VŠB TECHNICKÁ UNIVERZITA OSTRAVA
FAKULTA ELEKTROTECHNIKY A INFORMATIKY

Detekce anomálií v chování řidiče

Anomaly Detection in Driver Behaviour

Tomáš Sikora

Diplomová práce

Vedoucí práce: doc. Dr. Ing. Eduard Sojka

Ostrava, 2021

Abstrakt

Práce se zabývá detekcí anomálií v chování řidiče. Nejprve popisuje technická úskalí problému detekce anomální akce ve videozáznamu a následně sumarizuje používané techniky detekce anomálií v datech obecně. Hlavní částí práce je návrh a experimentální realizace systému, který s využitím subsystému detekujícího klíčové body pózy člověka v obraze a hloubkových neuronových sítí rozlišuje anomální a normální akce řidiče. Vhodnost vybraných konfigurací architektury neuronové sítě je následně ověřena v několika experimentech.

Klíčová slova

detekce anomálií; detekce anomálií u řízení vozu; detekce anomálií v chování řidiče; klíčové body; hloubkové učení; neuronové sítě; autoenkodér

Abstract

The main topic of this work is the detection of anomalies in driver's behavior. It describes technical difficulties of the problem of anomalous actions detection in the video. It summarizes the main methodologies generally used in the anomaly detection area. The core part of the work is the design and the experimental realization of the system that uses subsystem for keypoint detection in the video and deep neural networks to distinguish normal and anomalous actions of the driver. Several experiments are performed to validate the usability of selected keypoint features and neural network architectures.

Keywords

anomalies detection; anomaly driver behavior detection; anomaly behavior detection; keypoints; deep learning; neural networks; autoencoder

Poděkování

Rád bych tímto poděkoval především svému vedoucímu, doc. Dr. Ing. Sojkovi za cenné rady a připomínky, které mi věnoval při tvorbě této diplomové práce.

Obsah

Seznam použitých zkratek a symbolů

LSTM	–	Long short-term memory
RNN	–	Recurrent neural network
ROC	–	Receiver Operating Characteristic
P/R	–	Precision/Recall
AUC	–	Area under Curve.

Seznam obrázků

Seznam tabulek

Kapitola 1

Úvod

Automobilismus je oblastí, která nepochybně změnila a výrazně urychlila rozvoj lidské společnosti. Je to prostředek neuvěřitelného ekonomického růstu a jeden z hlavních symbolů technologického pokroku. V posledních letech dochází ke značnému vývoji v oblasti automatického řízení a autonomních systémů. Hnacím motorem vývoje je především potenciál posunout celý obor novým, úspornějším, bezpečnějším a pohodlnějším směrem. Je možné klasifikovat pět úrovní automatizace řízení.

- Asistence řízení, kdy automatizace pouze pomáhá řídit vůz, zvyšuje bezpečnost a usnadňuje a zrychluje rozhodování.
- Částečná automatizace, kdy některé prvky řízení zvládá automobil automaticky, ale zodpovědnost za řízení stále leží na řidiči, protože vyžaduje jeho neustálou pozornost.
- Podmíněná automatizace, kdy je automobil navíc schopný v ideálních podmínkách při malých rychlostech řídit téměř bez součinnosti řidiče.
- Vysoká automatizace, kdy automobil řídí bez pomoci řidiče, pokud to podmínky dovolují
- Plná automatizace, kdy je automobil plně autonomní.

Kategorií systémů spadajících do těchto úrovní jsou i podsystémy pro sledování aktivity řidiče. Jejich účelem je zaznamenání chování řidiče a jeho vyhodnocení. Pomáhají tak ostatním systémům na ně adekvátně reagovat. Toto je nezbytnou součástí přechodu mezi druhou a třetí, respektive třetí a čtvrtou úrovní automatizace. Je totiž důležité vědět, kdy se řidič věnuje řízení a kdy ne. Jde také o důležitý prvek bezpečnosti vozidla. Může totiž detekovat například zdravotní problémy.

Tato práce se zabývá popisem a návrhem právě takovéhoto systému. Jako vhodný prostředek pro realizaci řešení je beze sporu strojové učení. Bohužel je zde několik poněkud problematických oblastí. Jde především o dostupnost dat pro některé vzácné situace, například když u řidiče dojde k srdečnímu selhání nebo jiným potenciálně fatálním stavům. Pořizování a následná projekce videozáznamů z podobných situací může být vnímáno jako krajně nemorální. Nedá se proto očekávat, že by se

tato situace změnila vlivem technologického pokroku a modernizace monitorovacích systémů. Další poměrně závažný problém představuje generalizace klasifikace akcí samotných. Nelze zajistit vždy stejné výchozí podmínky. Tatáž akce může být jednou vyhodnocena jako problematická, jindy jako zcela normální.
Pro potřeby práce je detekce vzácných akcí zkoumána pouze v rozsahu vstupní datové sady. Přestože jde o velice zásadní aspekt, problém generalizace je řešen pouze okrajově.
Existuje několik různých druhů řešení. Jak již bylo řečeno, klasifikace vzácných akcí je v tomto případě v podstatě nemožná. K problému je tedy vhodné přistupovat jako k problému detekce anomálií. Vstupem systému je kamerový záznam, který bude sloužit jako základ pro extrakci série obrazových příznaků. V takto extrahované sérii je pak pomocí vhodného algoritmu provedena detekce anomálních situací.
Generalizace řešení pro jiné řidiče, světelné podmínky a systémové konfigurace není smyslem této práce. Pro potřeby této práce je kamerový záznam pořízen pouze z jednoho úhlu, z jedné kamery a v 2D prostoru.
Záznamy jsou pořízeny s několika figuranty předstírajícími anomální akce. Dále navrhovaný systém by tedy bylo nejdříve nutné personalizovat pro každého dalšího řidiče. Detekce anomálií je v tomto případě klasifikací binární. Systém nerozeznává, jestli jsou některé akce "anomálnější" než druhé nebo jestli je daná akce závažná. Systém také nerozpoznává jednotlivé typy akcí a kašel, třes nebo telefonování vyhodnocuje pouze jako anomálie.

Práce je členěna do osmi hlavních částí:

- **Kapitola 2** popisuje obecně celý problém a také již existující řešení pro detekci anomálií.
- **Kapitola 3** popisuje pre-rekvizity využité v samotném řešení. Cílem této kapitoly je zjednodušit orientaci v nadcházejících částech práce.
- **Kapitola 4** se zabývá základním navrhovaným řešením, jeho definicí a popisem
- **Kapitola 5** se věnuje experimentální realizací základního navrhovaného řešení včetně jeho výsledků a z nich vyplývající diskuse.
- **Kapitola 6** se zabývá vylepšeními základního řešení pomocí využití alternativních příznaků.
- **Kapitola 7** je zaměřena na realizace experimentálního vylepšení z předešlé kapitoly včetně jeho výsledků a z nich vyplývající diskuse.
- **Kapitola 8** popisuje vylepšení architektury, které vychází ze základního navrhovaného řešení.
- **Kapitola 9** je popisem experimentální realizace předešlé kapitoly a srovnáním jednotlivých vylepšení architektury navrhovaného základního řešení.

Cílem této práce je popsat problematiku detekce anomálií ve videozáznamu, především popsat specifické části problému samotného a jeho možné experimentální technické řešení pomocí algoritmů neuronových sítí a hloubkového učení. Práce se nezabývá specifickými problémy u sběru dat nebo jejich zpracování a nevěnuje se problémům plynoucím z nasazení systému do provozu.

Kapitola 2

Popis problému

Tato kapitola se zabývá obecným popisem systému sloužícího k detekci anomálií z videozáznamu. Nejdříve je popsán problém samotný a jeho realizace jako detekci anomálií bez učitele. Následuje popis jeho vstupů a výstupů spolu se základním popisem anomálie. Kapitola pokračuje částí věnovanou obecnému popisu přístupu pro detekci anomálií. Je rovněž uveden popis vybraných současných řešení.

2.1 Klasifikace akcí řidiče automobilu

Jak bylo uvedeno výše, existuje několik hlavních problémů klasifikace chování řidiče během řízení automobilu. Jistá podskupina činností, jako je volání, či jisté druhy nepozornosti, je dobře popsatelná vhodně sestrojenou datovou sadou. Bohužel toto je výrazně obtížnější pro situace způsobené zdravotními problémy, kdy je sběr dat nejen složitý, ale i krajně nemorální. V datové sadě tedy vzniká nevyrovnanost pro jednu z nejdůležitějších tříd chování. Je velice obtížné realizovat řešení problému pomocí přístupu učení s učitelem. Alternativou je přistupovat k němu jako k binární klasifikaci anomálií a rozdělit chování pouze na normální a anomální. Klasifikaci je možné provést s využitím algoritmů pro detekci anomálií a realizovat jako problém učení bez učitele.

2.2 Vstupy a výstupy

Vstupem problému je obrazový signál, v tomto případě videozáznam, pořízený z jednoho úhlu ve dvou dimenzích, který zaznamenává řidiče ve standardních a nestandardních situacích. Pro správnou klasifikaci akcí v automobilu je potřeba zvolit polohu kamery tak, aby snímala horní polovinu těla řidiče. Chování případných dalších osob v kabině vozidla není podstatné. Výstupem problému je vektor klasifikací, vyjadřující jestli video v daném časovém okně zaznamenává anomálii či nikoli. Protože je klasifikace binární, výstupem každé jednotlivé klasifikace je pouze 0 nebo 1 určující prezenci anomálie.

Obrázek 2.1: Použitý vhled kamery ve 2D

Vstupní videa lze rozdělit na:

- Videa obsahují anomálie, které jsou anotovány a přesně ohraničeny, aby bylo možné vyhodnotit jejich detekci.
- Videa zachycující normální/standardní chování, aby se dalo vyhnout případným shodám s anomálními daty.

Za anomálii je v tomto případě považováno jakékoli chování, rozpoznatelné pomocí klasifikačního systémů, které je v rozporu s bezpečným řízením vozu, nebo indikuje snížení pozornosti řidiče. Anomální videa jsou videa hraná, částečné za reálného řízení vozu a částečně s využitím simulátoru. Nejedná se tedy o stoprocentní reprezentaci reality. S její pomocí však lze validovat principy a fungování systému.

Nyní následuje popis základních principů využívaných pro detekci anomálií. Jde pouze o obecný popis v současnosti využívaných metod, založených na strojovém učení a také výčet reprezentantů jednotlivých metod. Samostatná (sub)kapitola je věnována metodám využívajícím neuronové sítě. Nejen proto, že v této oblasti dochází k dynamickému vývoji, ale také proto, že neuronové sítě jsou součástí návrhu řešení

2.3 Obecná sumarizace metod používaných pro klasifikaci anomálií

Detekce anomálií je pod-oblastí oboru datové analýzy zabývající se identifikací vzácných objektů, událostí nebo pozorování, které se významně liší od většiny dat [1]. Za anomálie se dá považovat odlehlé hodnoty, (v anglosaské literatuře označované jako outliters) data neviděná, (označovaná jako novelty) šum, ale také odchylky od očekávaných dat.
Algoritmy sloužící ke klasifikaci anomálií lze rozdělit do několika skupin dle jejich funkcionality. Dají se vypozorovat především tři hlavní proudy principů, které algoritmy využívají k tomu, aby anomálie odhalily.

- Detekce na základě rekonstrukce komprimuje vstupní sadu do její nízko-dimenzionální reprezentace a snaží se o její rekonstrukci zpět do původní podoby. V případě anomálních dat bude rekonstrukční chyba vykazovat velké hodnoty, protože algoritmus z jejich malého počtu (nebo úplné absence v tréninkových datech) není schopen vyhodnotit, jak správně rekonstruovat jejich podobu.
- Metody založené na hustotě a pravděpodobnostní distribuci dat, které se snaží aproximovat distribuci/hustotu vstupu a vybrat data, výrazně se vymykající distribuci či hustotě většiny datové sady. Do této kategorie se dají zařadit také metody fungující na principu srovnání vzdálenosti, jelikož se ve většině případů takto definované algoritmy dají vyjádřit pomocí hustoty nebo distribuce pravděpodobnosti a jejich principy bývají velice podobné.
- Třetí skupinou jsou metody rozdělující data na regiony. To provádí pomocí klasifikace normálních dat, což vylučuje data anomální.

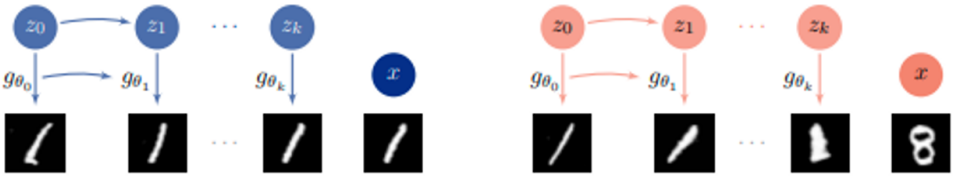
2.4 Klasické metody detekce anomálií

Mezi klasické metody se dají zařadit v podstatě všechny metody nevyužívající hloubkového učení a neuronových sítí. Tyto algoritmy často nepotřebují velkou normální datovou sadu, což je jedna z jejich výhod. Bývají také zpravidla rychlejší než hloubkové neuronové sítě. Nevýhody jsou velice často specifické pro jednotlivé algoritmy. Nejčastěji uváděným problémem je však jejich snižující se účinnost vzhledem ke zvyšující se dimenzionalitě dat. Také je nutné podotknout, že klasické metody jsou až na výjimky základem pro metody využívající hloubkových neuronových sítí. Je dobré tyto metody znát, aby bylo možné opravdu do detailu pochopit, jak tyto algoritmy fungují.
Hojně používaným přístupem pracujícím na principu rekonstrukce vstupu jsou metody založené na maticových operacích. Jako příklad lze uvést PCA [2] a především RPCA [3]. Algoritmus rozdělí vstupní matici X na matici s nízkou hodností L a řídkou matici S. To znamená, že při vhodném nastavení parametrů algoritmus separuje nízko dimenzionální reprezentaci datové sady do matice L a anomálie do matice S.
Dalším hojně používaným přístupem je využití shlukovacích algoritmů, [4] [5] [6] kdy anomální data nejsou součástí hlavních shluků. Je důležité si uvědomit, že tyto algoritmy jsou ve své podstatě

založeny na principu aproximace hustoty pravděpodobnosti. Časté a používané řešení je v tomto případě i využití klasifikačních algoritmů jako je K-NN [7], kdy nejde o dělení dat na specifické regiony, ale spíše o shlukování dat sobě blízkých, v tomto případě anomálních.
Pro úplnost je nutné uvést ještě klasifikaci pomocí SVN [8], tzv. jedno-třídové klasifikace (one-class learning), kdy SVN během tréninku dokáže vytvořit hranici pro rozlišení majoritní třídy, tedy normálních dat a třídy minoritní – anomálií. Tento přístup je nejefektivnější, pokud anomální data postrádají hierarchii a jsou roztroušena mezi daty normálními. Algoritmem ze stejné skupiny, ale pracujícím na opačném principu, je pak Isolation Forest [9]. Jak jeho název napovídá, využívá předpokladu, že bude snazší izolovat anomální než neanomální data.

2.5 Metody využívající hloubkových neuronových sítí

V současné době jsou algoritmy pro detekci anomálií často realizovány pomocí neuronových sítí. Takto realizované klasifikátory mají zpravidla vyšší úspěšnost, která je ale úzce vázána na kvalitu a množství tréninkových dat. Toto není nepřekonatelný problém a poměrně efektivním řešením je před-trénování, kdy i použití pouze vzdáleně podobné datové sady zvyšuje úspěšnost u klasifikace o jednotky až desítky procent. Jak je naznačeno výše, tyto algoritmy fungují velice často na obdobných principech jako algoritmy klasické, nebo je přímo rozšiřují.
Snad nejznámějším přístupem je využití autoenkodér architektury, který je detailněji popsán později v kapitole 3.2. Na tomto místě je vhodné zmínit některé metody tvořené kombinací této architektury s jinými přístupy. Jedním z nich je kombinace s attention modulem A3D [10], kdy je místo kontextového vektoru použit Luong-Attention [11]. Dalším poměrně rozšířeným přístupem je DAGMM [12], který je kombinací autoenkodéru a gausovského smíšeného modelu (Gaussian mixture model).
Jednou z implementací využívající attention je sBiLSAN [13], kdy je použita dvousměrná LSTM síť zakončená vícehlavými self-attentnion vrstvami. Tento přístup vykazuje výrazně lepší výsledky než standardní LSTM sítě určené ke klasifikaci anomálií, nicméně princip self-attention vrstev je poměrně nový, a zatím tedy není příliš rozšířený.
Poměrně unikátním přístupem je využití tzv. generativních sítí nebo také GANZ. Rád bych uvedl například metodu ADGAN [14], která pracuje s myšlenkou dvou neuronových sítí, generátoru a diskriminátoru, které pracují s protichůdnými cíli. Generátor generuje data vzorkovaná náhodně z nor-



Obrázek 2.2: Rekonstrukce dle ADAGAN [14]

málové distribuce a na základě vah se snaží vygenerovat vzorek, který není rozeznatelný od tréninkové sady. Diskriminátor naproti tomu dostává jak tato data, tak data z tréninkové sady a snaží se rozpoznat, jestli jsou pravá nebo ne. Tímto se postupně oba zlepšují ve svých úkolech a generátor se učí generovat data co nejpodobnější distribuci tréninkové sady. U inference se využívá především generátor. Pro klasifikaci anomálnosti/normality dat je použit gradientní sestup, aby spočetl inverzní, nízko-dimenzionální reprezentaci vstupu, která je pak použita pro generaci nového výstupu. Následně je použito L2 skoré pro klasifikaci vzdálenosti vygenerovaného obrazu a tím pádem klasifikace úrovně jeho normality. Pokud je generátor schopen vygenerovat data podobná těm vstupním, pravděpodobně nepůjde o anomálie.
Pro úplné a správné pochopení navrhovaného řešení je níže v textu uveden popis pre-rekvizit.

Kapitola 3

Algoritmus pro detekci anomálií

Pro pochopení řešení je vhodné nejdříve poznat několik pre-rekvizit, použitých v jeho částech. Snažím se o poměrně podrobný popis, který by měl poskytnout základ k pochopení zbytku práce.

Jedná se o:

- Systém pro přípravu obrazových příznaků – Open-Pose.
- Algoritmus pro detekci anomálie a jeho vylepšení – autoenkodér, attention.
- Typ rekurentních buněk neuronové sítě – LSTM.
- Vylepšení tréninkového procesu – normalizaci vrstev a inicializaci vah.
- Algoritmus určující hranici detekce anomálie – ROC křivku.

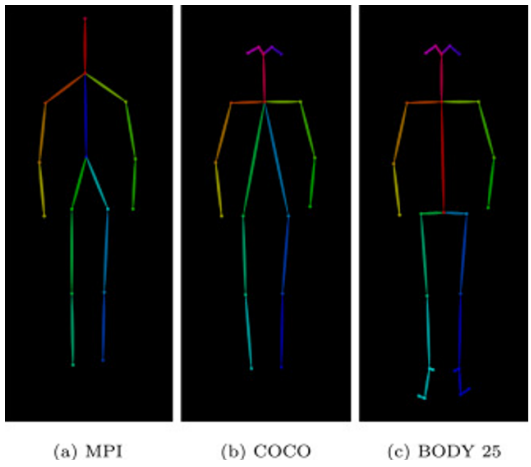
3.1 Systém pro přípravu obrazových příznaků

Jak bylo již řečeno v kapitole 2.2, činnost řidiče je zaznamenána videem. Podle záznamů je možné vypozorovat jisté chování, které jde na základě analytického procesu klasifikovat jako anomální nebo normální. Je vhodné analyzovat pouze část obrazu, týkající se přímo řidiče. Toto následně zjednoduší proces a ulehčí práci algoritmu vyhodnocujícímu výslednou anomálnost. Cílem je nejdříve v obraze nalézt sérii příznaků a dále pracovat pouze s nimi. Existuje mnoho takovýchto přístupů, jednak klasičtější metody jako je HAAR[15] HOG[16], LBP[17] a jim podobné. Metody detekující klíčové body jako je například SIFT[18] nebo SURF[19] a alternativou je samozřejmě také využití klasických konvolučních neuronových sítí[20].
Algoritmus by měl pracovat s co nejmenším a nejreprezentativnějším vstupním vektorem, čímž se výrazně snižuje komplexita detekce. Je tedy vhodné využít příznaky tvořené klíčovými body, protože jejich dimenzionalita je v tomto případě velice nízká. Body budou charakterizovat pozice jednotlivých částí těla řidiče. Je ale poměrně složité vytvořit klasifikátor využívající klasické přístupy jako je SIFT nebo SURF tak, aby v omezeném prostoru a silně nestabilních podmínkách automobilu fungoval bez jakýchkoli problémů.

Alternativou je detekce pózy řidiče pomocí neuronových sítí. Jednou z prvních prací zabývající se tímto problémem byl Deep Pose[21] firmy Google. Tento systém využívá konvolučních neuronových sítí a v době svého uvedení představoval revoluci v této oblasti. Algoritmus vyhledává klíčové body, odpovídající kritickým částem těla. Z nich se dá vytvořit obraz pózy člověka. Pomyslným následovníkem tohoto přístupu (alespoň vzhledem k principiálnímu využití konvolučních neuronových sítí) je i systém Open-Pose [22].

Obrázek 3.1: Klíčové body s využitím Open Pose[22]

Open Pose vznikl v roce v 2018 v CMU Perceptual labu. Systém se hodí pro detekci klíčových bodů v reálném čase. Umožňuje zpracování videa, kdy je velice jednoduché z výstupu systému získat video s klíčovými body nebo pokud to není třeba pouze vektor obsahující klíčové body samotné. Velkou výhodou je také schopnost detekce pózy hned několika lidí zároveň. Open Pose je schopen krom rekonstrukce pózy ve 2D pracovat s několika kamerami a provádět rekonstrukcí ve 3D.
Konstrukci pózy je možné reprezentovat jedním ze tří modelů, jsou to MPI, COCO a BODY_25. Jednotlivé modely ve liší úspěšností detekce, pozicí či dostupností některých bodů a v neposlední řadě rychlostí konstrukce modelu samotného. V době vzniku této práce dosahoval nejlepších výsledků model BODY_25, to jak v přesnosti detekce, tak celkovou rychlostí konstrukce.



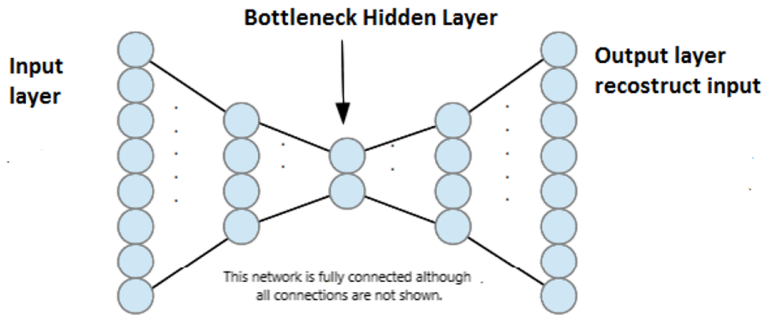
Obrázek 3.2: Modely používané Open Pose[23]

BODY_25 se skládá ze 25ti klíčových bodů, ale během jízdy automobilem připadá v úvahu pouze 15 z nich. Tyto body reprezentují horní polovinu těla. Model je možné parametrizovat tak, aby na úkor rychlosti extrakce docházelo k přesnější lokalizaci jednotlivých klíčových bodů. Nyní, kdy je známa přibližná podoba dat, je možné přejít k algoritmu využitému pro samotnou detekci anomálií v datech.

3.2 Algoritmus pro detekci anomálie

V kapitole 2.3 je popsána skupina algoritmů sloužících k detekci anomálií. Jak vyplývá z kapitoly 2.5, pro potřeby této práce byl využit algoritmus autoenkodér. Autoenkodér je architektura neuronové sítě, která se využívá pro detekci anomálií pomocí principu rekonstrukce vstupního vektoru. Autoenkodér dává dostatek prostoru pro experimentaci, je vcelku dobře popsán a jeho základní parametry jsou velice dobře známé.

Jako algoritmus pracující na bázi rekonstrukce vstupního vektoru, se autoenkodér snaží komprimovat tento vektor do reprezentace o menší dimenzi. Takto komprimovaný vstup se pak autoenkodér snaží rekonstruovat do původní podoby. Pokud je neuronová síť trénovaná pouze s využitím normálních dat, a tréninkový proces proběhne korektně, budou data podobná těm normálním rekonstruována s malou rekonstrukční chybou. V případě, kdy autoenkodér zpracovává pro něj dosud neznámá data, komprimace reprezentace není příliš přesná a jeho rekonstrukční chyba je v tomto případě velká. Tato data se pak dají označit jako anomálie.

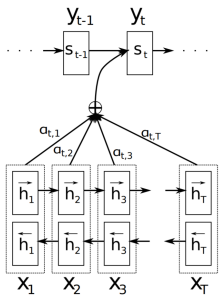


Obrázek 3.3: Struktura autoenkodéru [24]

Z pohledu architektury neuronové sítě je autoenkodér velice jednoduchý a přímočarý. Dá se rozdělit na dvě části. Enkodér komprimující sadu vstupů do vektoru velikosti nejmenší vrstvy, zpravidla uprostřed celé sítě. Dále pak dekodér, který tuto representaci dekóduje a snaží se tedy rekonstruovat zpět sérii vstupů. Obě části bývají většinou stejného počtu vrstev a obě struktury se zrcadlí. Počty jednotek bývají zpravidla klesající směrem k úzkému hrdlu a stoupající směrem ke vstupům/výstupům.

3.2.1 Rozšíření implementace algoritmu detekce anomálií

Pokud bude vstupem a výstupem autoenkodéru sekvence, u klasické implementace je na konci enkodéru nutné komprimovat všechny kroky vstupní sekvence, prostřednictvím posledního skrytého stavu do jediného vektoru, který bude předán dekodéru. To může snižovat úspěšnost rekonstrukce, vytvářet na výstupu šum a zvednout hodnoty rekonstrukční chyby pro normální data, respektive snížit úroveň rekonstrukční chyby pro data anomální. Jedním ze způsobů, jak se tomuto efektu vyhnout, je zvětšení počtu jednotek neuronové sítě, což ale nemusí být vždy žádoucí a snižuje celkovou efektivitu komprese. Proto je v experimentu 8.3 zaměněno úzké hrdlo enkodéru za modul attention, (pozornosti) který se v poslední době stává poměrně rozšířenou součástí mnoha implementací sekvenčních modelů. Hlavním záměrem je, aby v některých případech dekodér věnoval zvláštní pozornost pouze některým specifickým částem vstupní sekvence.
V současné době existuje několik implementací attention mechanismu. Vzhledem k omezenému rozsahu práce bude dále popsán pouze princip tzv. Bahdanau-Attention[25], protože je nejjednodušší a principiálně velice podobný ostatním. Vstupu dekodéru je místo komprimovaného výstupního vektoru sekvence enkodéru předán vektor pevné délky, reprezentace všech skrytých stavů enkodéru. V tomto případě jde o jejich součet. Takto vytvořený vektor je ještě parametrizovaný klasickou neuronovou sítí. To může znít poněkud složitě, ale v důsledku jde vlastně o lineární kombinaci vstupních a výstupních skrytých stavů. Attention vektor určuje, jak velkou by měl dekodér věnovat pozornost jednotlivým prvkům vstupní sekvence. V této práci je také replikována architektura již existujícího výzkumu A3D[10], který vychází z tzv. Luong[11] attention.

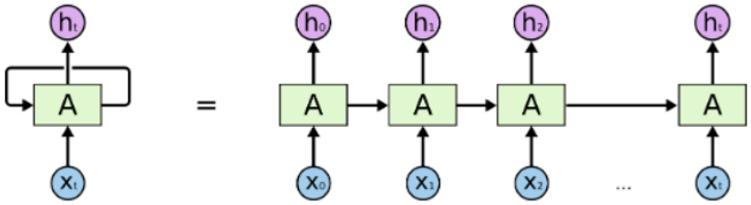


Obrázek 3.4: Bahdanau-Attention mechanismus[25]

Autoenkodér se realizuje běžně pomocí fully-conected vrstev, což v tomto případě není ideální. Může to způsobovat problémy během tréninku a komplikovat nastavení některých hyperparametrů. Pro detekci anomálie je vhodné používat celé série příznaků. Je totiž možné narazit na pózy, které jsou v jedné situaci normální a ve druhé anomální, což je v sekvenci za sebou daleko méně pravděpodobné. Tento nedostatek se dá vyřešit i pomocí standardních fully-connected vrstev. To se ale příliš nedoporučuje, protože tento postup není vhodný k zachycení delších sekvencí a negeneralizuje příliš dobře pro velké množství dat. Vhodnější alternativou je využití rekurentních vrstev, které jsou k tomu přímo určeny.

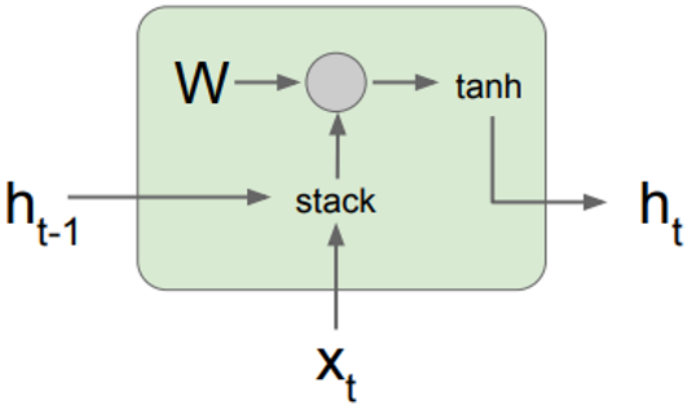
3.3 Realizace rekurentních spojení

Protože je cílem vyhledávat anomálie v po sobě jdoucích časových řadách, je vhodné realizovat neuronovou síť s využitím rekurentních jednotek. Rekurentní neuronové sítě byly poprvé popsány v roce 1990[26]. Pro sekvenci vstupů o indexech *1, 2, . . . , t-1, t,* je rekurentní neuronová síť schopna zachytit časové závislosti a representovat je ve výstupní dávce. Provádí to spojením výstupu z času *t-1* na vstup jednotky v čase *t*. Každá rekurentní jednotka má skrytý vnitřní stav, který je závislý na stavu předchozím.



Obrázek 3.5: Rekurentní neuronová síť v čase [27]

Vnitřní reprezentace rekurentní buňky je velice podobná standardní fully-connected vrstvě. Rozdílem je již zmíněné spojení předchozího stavu a jeho sloučení s aktuálním vstupem, jak je vidět na obrázku 3.5. Trénink probíhá standardním způsobem, nicméně jak je z principu sítě zřejmé, propagace gradientu chyby je prováděna skrze čas. Pro výpočet chyby gradientu z posledního kroku je třeba nejdříve vypočítat gradient pro kroky následující. To znamená, že doba tréninku rekurentních neuronových sítí se zvyšuje s počtem kroků v dávce a jeho paralelizace je poměrně problematická. Velkou výhodou zůstává schopnost rekurentních jednotek udržet kontext skrze vstupní dávku. Jednou z méně příznivých vlastností rekurentních jednotek je exploze či zmizení gradientu. Proto se v praxi klasické rekurentní buňky až na výjimky nepoužívají.



Obrázek 3.6: Struktura RNN buňky [28]

U dlouhých vstupních sekvencí dochází k explozi nebo zmizení gradientu. Při pohledu na obrázek 3.5 je zřejmé, že je možné brát každý krok sekvence jako samostatnou jednotku. U každého kroku propagace dochází k monotónnímu násobení stejné matice. Pro zjednodušení je dobré si představit matici W jako racionální číslo. Pokud je gradient chyby ostře větší než jedna, bude v každém dalším kroku větší a cestou ke vstupu bude jeho hodnota teoreticky příliš velká a nepoužitelná pro pokračování tréninkového procesu – gradient exploduje. Pokud je gradient ostře menší než jedna, cestou ke vstupu se bude zmenšovat. Je také možné, že signál chyby takto úplně zmizí. Naštěstí, tento problém dokáží vyřešit modernější implementace rekurentních jednotek popisované dále, jako jsou LSTM[29] nebo GRU[30].

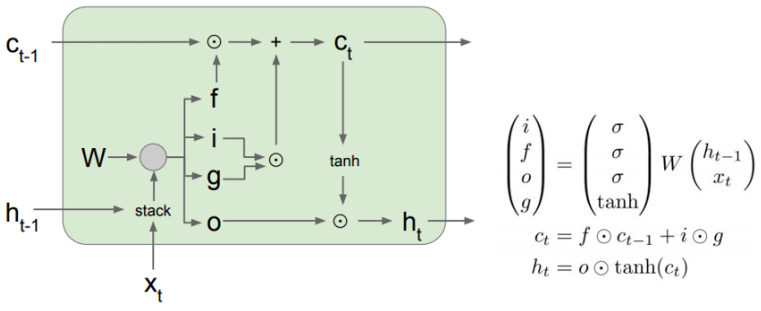
3.3.1 Vylepšené řešení rekurentních spojení

LSTM[29] je navrhnuto jako řešení problému explodujícího/mizejícího gradientu chyby, respektive problému zpracování časových závislostí v delších vstupních sekvencích. Základní myšlenkou je schopnost zapomenout některé části vstupní sekvence a soustředit se pouze na vybrané části. Dále bude zohledněna pouze architektura standardní LSTM a její klasická implementace popsaná ve studii uvedené výše. Princip fungování je poněkud složitější než je tomu u klasického RNN.

LSTM buňky mají kromě skrytého stavu h (který je zároveň výstupem buňky pro daný čas t) i stav buňky c. Další změnou implementace je řízení buňky třemi tzv. gate spojeními (brána), kterým předchází matice vah w, jejíž rozměry jsou čtyřnásobkem rozměru předchozího skrytého stavu (toto se v některé literatuře zobrazuje jako váha před každou gate, nicméně pro pochopení, je vhodnější si představit tyto váhy jako celek).

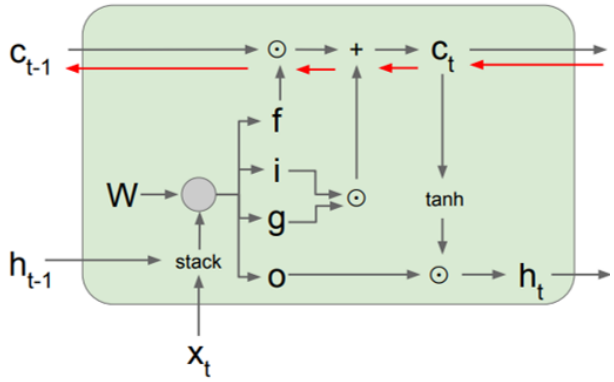
Gate spojení slouží k regulaci toku následujícím způsobem:

- Forget gate (f) – zodpovědná za zapomenutí informací z předchozího stavu.
- Input gate (i) – rozhoduje, co budeme updatovat ve stavu buňky.
- Gate gate (g) – určuje, jak moc informace bude zapsáno do stavu buňky.
- Output gate (o) – rozhoduje které z částí stavu LSTM půjdou na výstup.



Obrázek 3.7: Vnitřní architektura LSTM buňky[28]

Forget, Input a Output gate používají sigmoidovou aktivační funkci, čímž omezují výstup do intervalu *(0,1)*. Je možné je tedy chápat jako jakési filtry rozhodující o tom, jak velká část jimi násobeného signálu bude dále přenesena. Výpočet výsledku je pak přímočarý, hadamardovým násobením Forget gate a předchozího stavu buňky dojde k redukci informace a jejímu "zapomenutí". Nový vnitřní stav buňky je pak součtem stavu předchozího a hodnoty Gate gate, která je opět limitována dalším hadamardovým součinem s input gate. Pro výpočet skrytého stavu je pak výstup vnitřního stavu pomocí *tanh* funkce převeden do rozmezí *(-1,1)* a následně je opět pomocí Output gate hadamardovým násobením rozhodnuto, jak velká část bude nakonec použita.



Obrázek 3.8: Propagace gradientu chyby v LSTM [28]

Problém mizejícího a explodujícího gradientu je velice elegantně potlačen. Vnitřní stav c prochází během zpětné propagace pouze jednou derivací nelineární funkce *tanh*, a to hned v prvním kroku. Jeho další propagace je nyní v nejhorším případě negativně ovlivněna pouze hadamardovým násobením s Forget Gate, jehož sigmoidová aktivanční funkce je poměrně stabilní, navíc bude vnitřní stav násoben potencionálně různými hodnotami Forget gate a není monotónním násobením celé matice jako u klasické RNN. Metoda zpětné propagace vnitřního stavu je velice podobná residuálním spojením u neuronové sítě RESNET [31]. Gradient je pak daleko lépe propagován pro váhovou matici w, což zajišťuje daleko lepší zachycení souvislostí v delších sekvencích.

V základní implementaci bylo proto použito LSTM, je to jednoduchý, dobře popsaný algoritmus. Je velice flexibilní a poskytuje mnoho možností nastavení, týkajících se především dimenzí vstupní dávky, její délky a dimenze výstupů. To bohužel neznamená, že k explozi/zmizení gradientu nemůže dojít a v praxi se to stává vlastně poměrně často. Během experimentů se vyskytlo několik situací, kdy gradient chyby opravdu explodoval. Naštěstí existuje několik řešení, které tuto situaci dále zlepšují a zvyšují stabilitu sítě a tréninku, kterým se dále věnuji v kapitole 3.4.

3.4 Prvky zvyšující stabilitu tréninku sítě

Neuronové sítě jsou algoritmem, u nějž se často zapomíná na roli náhody. Na začátku každého tréninku jsou váhy každé z vrstev nastaveny na náhodnou hodnotu určenou inicializačním procesem. Tento přístup může za ideálních okolností fungovat bez jakýchkoli potíží. Je ale možné, že vlivem náhody dojde k destabilizaci, jako je například exploze nebo zmizení gradientu, či zpomalení tréninku. To se stává zpravidla ze dvou důvodů. Za prvé, u inicializace vah může dojít ke stavu, kdy hodnoty jednotlivých vah, jsou příliš velké nebo naopak malé. Gradient chyby pak během optimalizace vlivem nelinearity konverguje extrémním hodnotám. Druhou možností pak je, že se v datové distribuci nachází extrémní hodnoty, které v jistém bodě destabilizují trénink. Tato kapitola popisuje možná řešení obou těchto problémů.

3.4.1 Způsob inicializace vah

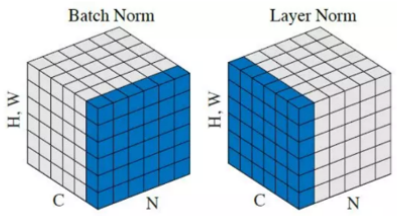
Jak bylo řečeno výše, jednou z příčin explodujícího gradientu může být nevhodná inicializace vah pro jednotlivé vrstvy či jednotky. Špatná inicializace může také způsobovat vyšší rozdíly ve výsledcích optimalizačního procesu nebo prodlužovat dobu nutnou pro trénink. Pro potřeby této práce bylo využito jednoho z dostupných dobře popsaných procesů – Xavierovu[32] inicializaci, která volí hodnoty vah z uniformní normální distribuce ohraničené vztahem.

$$\pm\frac{\sqrt{6}}{\sqrt{n_i + n_{i+1}}} \tag{3.1}$$

kde n_i je počet vstupů do vrstvy a n_{i+1} je počet jejích výstupů.

3.4.2 Normalizace dat

Ve tréninkových datech se často objevují extrémní případy, kdy propagace gradientu jejich chyby může způsobit zmíněnou nestabilitu tréninku. To se projevuje jak větší šancí na explozi nebo zmizení gradientu, tak celkovým zpomalením. Namísto klasické normalizace dávky, která se v případě rekurentních vrstev zpravidla nedoporučuje, byla zvolena normalizace vrstvy. [32]. Normalizace vrstvy je varianta normalizace dat na úrovni jednotlivých vrstev neuronové sítě místo normalizace na úrovni jednotlivých dávek.



Obrázek 3.9: Srovnání normalizace vrstvy a dávky[33]

3.5 Algoritmus určující hranici detekce anomálie

Výstupem algoritmu autoenkodéru je rekonstruovaná reprezentace vstupního vektoru. Úroveň chyby je pak možné vyjádřit jako výsledek srovnání vstupu a této hodnoty pomocí vhodné chybové funkce. Pokud bude tato hodnota vysoká, jedná se o data anomální, pokud bude nízká, jde o data normální. Rozlišení prahové hodnoty, kdy lze říci, nakolik je chyba vysoká a zda už se jedná o anomálii, je samostatným a téměř optimalizačním problémem. Jedním ze způsobů, jak tuto hranici analyzovat, je ROC křivka. ROC křivka je nástroj, který popisuje vlastnosti binárního klasifikátoru v závislosti na různých nastaveních jeho klasifikačního prahu. ROC křivka byla původně vyvinuta kolem roku 1950 pro analýzu radarového signálu a jeho odlišení od šumu.
Ve chvíli, kdy je proveden výběr konkrétní hranice a jsou klasifikovány hodnoty, je možné vytvořit pravdivostní matici obsahující pravé pozitiva (TP), falešná pozitiva (FP), pravá negativa (TN) a falešná negativa (FN). Cílem je, aby byla klasifikace co nejpřesnější, tedy aby pravá pozitiva (správně klasifikované anomálie) převažovala nad falešnými pozitivy (špatně klasifikované anomálie). ROC křivka nabízí možnost, jak porovnat různé hodnoty hranice, a tedy takto vytvořené matice, mezi sebou a následně vybrat vhodnou hodnotu hladiny na základě požadovaných vlastností klasifikátoru.

Pro každé jednotlivé nastavení hranice je nutné spočíst:

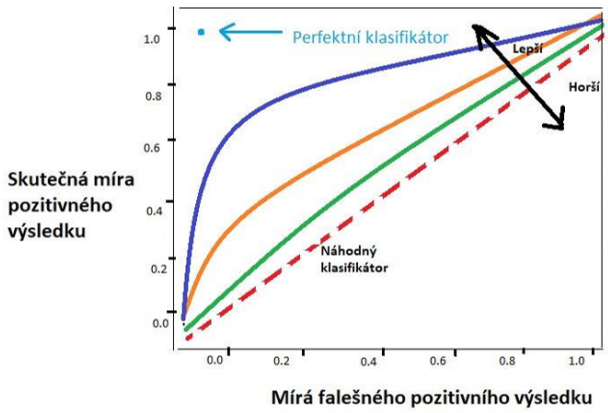
- TPR (True positive rate), které je počítáno stejně jako citlivost – počet pozitiv vyděleno součtem pozitiv a falešných negativ.
- FPR (False positive rate), což je obrácená hodnota specificity – počet falešných pozitiv vydělen počtem falešných pozitiv a pravých negativ.

$$TPR = TP/(TP + FN) \tag{3.2}$$

$$FPR = FP/(FP + TN) \tag{3.3}$$

Tyto hodnoty je pak možné zanést do tzv. ROC grafu a porovnat charakteristiky klasifikátoru s využitím tzv. AUC hodnoty. AUC hodnota vyjadřuje pravděpodobnost schopnosti modelu správně odlišit příslušnou anomálii, což znamená, že dokáže rozdělit signál od šumu. Graficky je znázorněna jako plocha pod křivkou.

- AUC 1 znamená ideální klasifikátor.
- AUC 0.5 znamená, že klasifikátor je v podstatě náhodný.
- AUC 0 znamená klasifikátor klasifikující vždy špatně.



Obrázek 3.10: ROC křivka v grafu

Algoritmus ROC křivky byl zvolen, protože dovoluje vybrat optimální hladinu chyby tak, aby docházelo k co nejmenšímu počtu falešných pozitiv, což je v tomto případě jeden z nejdůležitějších ukazatelů klasifikátoru. Je velice důležité, aby objevené anomálie byly skutečnými anomáliemi, jelikož každá falešná detekce anomálie by snižovala důvěru uživatele v tento systém. Samotný návrh teoretických vlastností produkční verze není předmětem této práce. Jde ale o důležitou vlastnost, kterou je potřeba brát v potaz. Další velkou výhodou je možnost srovnání jak TPR/FPR hodnot pro jednotlivé hladiny, tak rychlé srovnání klasifikátorů v rámci AUC hodnot.
V této práci je rovněž využit algoritmus Precission/Recall curve nebo také křivky přesnosti a citlivosti. Hodnota přesnosti vyjadřuje poměr správně detekovaných výsledků vzhledem k celkovému počtu detekcí. Hodnota citlivosti vyjadřuje naproti tomu počet správně detekovaných vzorů vzhledem k jejich celkovému počtu. Tyto hodnoty jsou použity obdobně jako TPR a FPR u standardní ROC křivky.

$$Precision = TP/(TP + FP) \tag{3.4}$$

$$Recall = TP/(TP + FN) \tag{3.5}$$

Po popsání nejdůležitějších pre-rekvizit je nyní možné přejít k samotnému základnímu řešení.

Kapitola 4

Základní navrhované řešení

Tato kapitola popisuje základní navrhované řešení problému detekce anomálního chování řidiče ve videozáznamu, jak bylo definováno v kapitole 2.1. Algoritmus nejdříve převede obrazový signál na sérii obrazových příznaků, což jsou v tomto případě klíčové body, jejichž detekce je blíže popsaná v kapitole 3.1. Klíčové body jsou následně transformovaný na vhodnější příznaky – úhly mezi vybranými body. Samotná realizace algoritmu zajišťujícího detekci anomálií je prováděna pomocí neuronové sítě využívající architektury autoenkodéru. Ten je detailněji popsán v kapitole 3.2. Jeho realizace, a především nastavení hodnot hyperparametrů je výsledkem několika komplexních prohledávání. Konečná implementace využívá rekurentních jednotek, v tomto případě LSTM, popsaných podrobněji v kapitole 3.3. V této kapitole se snažím popsat obecné vlastnosti a konstrukci řešení. Detailní popis jeho experimentální realizace, použité metodiky a jeho výsledků je pak uveden v následující kapitole.

Kapitola samotná je strukturována následovně:

- **Kapitola 4.1** popisuje základní konstrukci obrazových příznaků z obrazového signálu. Způsob realizace pomocí systému Open-Pose a následnou transformaci těchto příznaků na úhly, jejich popis, vlastnosti a výběr základní sady.
- **Kapitola 4.2** popisuje následnou konstrukci vstupní dávky, vlastnosti a možnosti její úpravy.
- **Kapitola 4.3** popisuje architekturu neuronové sítě, která dále vyhodnocuje jejich anomálnost.
- **Kapitola 4.4** popisuje návrh a postup tréningu, inference a vyhodnocení chyby.

4.1 Konstrukce obrazových příznaků

Tato kapitola pojednává o detekci a konstrukci série příznaků z obrazového signálu. Nejprve je popsán obrazový signál (video), výběr úhlu kamery a také obecné vlastnosti záznamu. Navazuje popis extrakce obrazových příznaků, což jsou v tomto případě klíčové body. Dále je nastíněna

problematika prezentace bodů s využitím úhlů, které je možné následně s jejich pomocí zkonstruovat. Je rozebrán rovněž způsob realizace převodu z příznaků tvořených body na úhly a definována jejich základní série, využita ve zbytku práce.

4.1.1 Obraz řidiče a jeho vlastnosti

Jak bylo řečeno v kapitole 2.2, vstupem tohoto problému je sekvence obrazových snímků, videozáznam ve 2D, ze kterých jsou extrahovány obrazové příznaky – klíčové body. Je nutné, aby záznam dosahoval určité minimální kvality pro extrakci a zároveň splňoval kritéria, díky kterým bude možné následně vytvořené sekvence analyzovat pomocí vhodného algoritmu.
Jednou z podmínek je stejný počet snímků za vteřinu u všech tréninkových i validačních videí. Pokud se tak nestane, není možné dobře zachytit časové závislosti jednotlivých příznaků. Barevnost a rozlišení videí nejsou pro účely této práce podstatné. Zásadním aspektem je vhodné umístění a natočení kamery. Vzdálenost není příliš důležitá, ale úhel definuje, které klíčové body budou vidět. Po převodu klíčových bodů na úhly (viz. níže) definuje natočení kamery přímo jejich hodnoty, vlastnosti a vztahy mezi nimi.
Vzhledem k charakteru problému je podstatné, aby bylo vidět ruce řidiče, jeho hlavu a sklon těla. Existují tři hlavní úhly kamery, viditelné na obrázku 1.14, které jsou využívány vědeckou komunitou a výrobci automobilů pro realizaci systémů záznamu řidiče.

Jde především o:

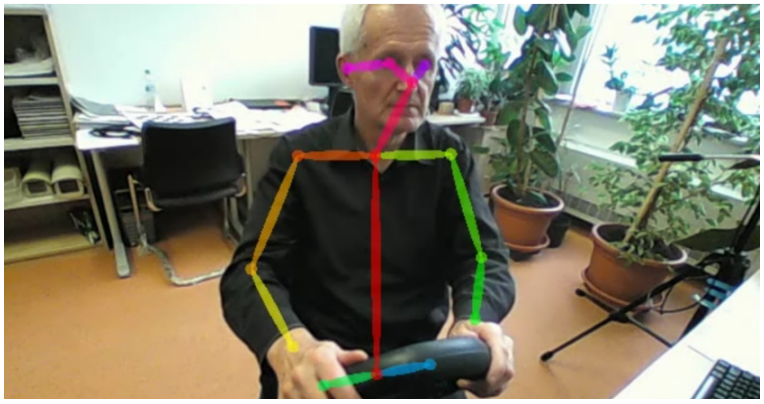
1. Pohled z boku – obrázek 1.14 - 1
2. Pohled z palubní desky – obrázek 1.14 - 2
3. Pohled ze zpětného zrcátka řidiče – obrázek 1.14 - 3

Obrázek 4.1: Vhledy kamery 1,2 a 3 pro záznam řidiče

Každý z těchto úhlů má jisté výhody a nevýhody, po experimentaci se ale ukazuje, že nejvhodnější variantou je pohled ze zpětného zrcátka (4.1 spodní obrázek). První varianta nezachycuje některé náklony hlavy a vhled kamery může být blokován/rušen spolujezdcem. V případě druhé varianty není vidět levá paže, což může znamenat úplnou neschopnost zachycení některých anomálních pozic. Pohled ze zrcátka je nejblíže k řidiči, což pomáhá při extrakci obrazových příznaků. Další výhodou je jednoduchost, se kterou je možné pouhým oříznutím obrazu izolovat řidiče a zjednodušit tak systému následnou detekci důležitých klíčových bodů.

4.1.2 Konstrukce sady příznaků

Konstrukce sady příznaků je prováděna pomocí systému Open-Pose. Ten detekuje klíčové body, které zobrazují pozici jednotlivých částí těla v obraze. Každý bod je definován svou 2-D souřadnicí. V případě, kdy není některá z částí těla vidět, se body nezobrazují a jejich souřadnice je neznámá. Jak bylo uvedeno v kapitole 3.1, v obrazovém signálu je detekováno 25 klíčových bodů. To ale během jízdy automobilem není potřeba, protože v úvahu připadá pouze horní polovinu těla, tedy 15 klíčových bodů. To vede k velkému dimenzionálnímu zjednodušení, protože není třeba pracovat s obrazem o rozměrech čítajících několik tisíc obrazových bodů, ale pouze s několika body ve dvou dimenzích. Velká část anomálních akcí se dá zaznamenat pouze několika příznaky. Ve většině případů bude vstupní dimenze ještě menší.



Obrázek 4.2: Klíčové body modelu BODY_25 v obraze

Jelikož každý klíčový bod určuje souřadnice jedné části těla v obraze, je pravděpodobné, že pokud si za volat sednou různí lidé, jejich pozice se budou lišit. Pro detekční algoritmus to znamená další úkol v podobě rozlišení anomálnosti a normálnosti stavů jedné osoby vůči druhé. Přidání další dimenze, ale výrazně komplikuje problém klasifikace. Byť jsou neuronové sítě teoreticky schopny s pomocí dostatku dat tento jev řešit, je vhodné je transformovat tak, aby nebyla závislá na absolutní pozici bodů na obrazovce.
Vhodným řešením jsou úhly mezi klíčovými body. Jejich konstrukce je poměrně intuitivní záležitostí. Nejdříve je nutné vybrat trojici klíčových bodů (trojúhelník), vyjadřující prostorovou závislost, kte-

rou má úhel sledovat. Může jít o anatomickou závislost, například úhel bodů sledujících rameno nebo o závislost plynoucí z polohy řidiče vozu – úhel mezi body tvořenými předloktím a nosem. Úhel se pak měří u středního, tedy druhého bodu každé trojice (měřeno ve stupních).

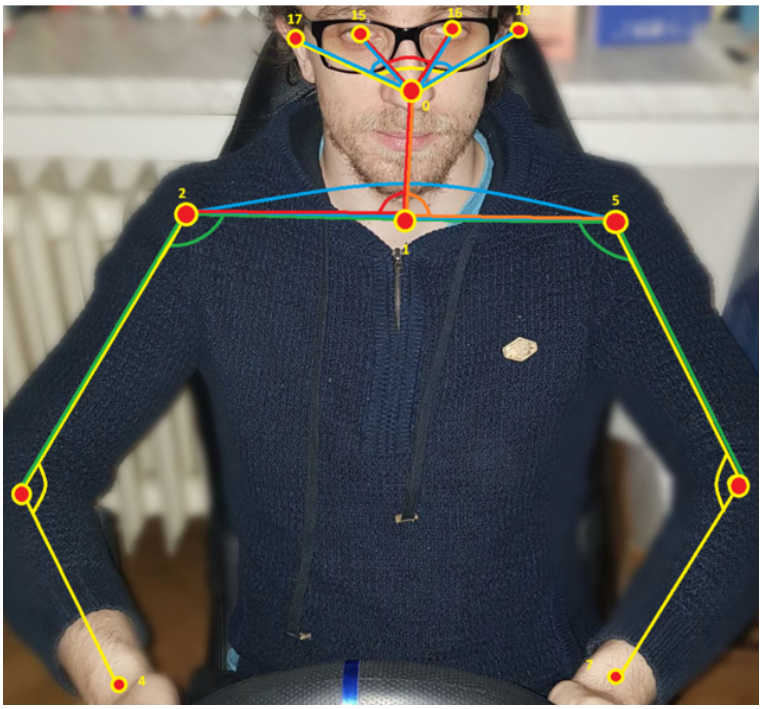
Výpočet je proveden například pomocí kosinovy věty dle vztahu:

$$c^2 = a^2 + b^2 - 2ab \times cos(\alpha) \qquad \alpha = arccos\left(\frac{a^2 + b^2 - c^2}{2ab}\right) \tag{4.1}$$

V navrhovaném základním řešení byla využita série úhlů v rozmezí 0°-180°, kdy není rozdíl mezi úhlem 25° a -25°. Vzhledem k poměrně malému rozsahu pohybu řidiče v automobilu se to jeví jako naprosto dostatečné. Zmenšení rozsahu pak výrazně ulehčuje klasifikaci anomálií algoritmem. Experimenty byly prováděny i s úhly v rozsahu 0°-360°. Toto se ale příliš neosvědčilo - přidání celých úhlů zvyšovalo šum, ale nezlepšovalo výsledky detekce.
I když je teoreticky možné využívat všechny trojice bodů, není to ve většině případů žádoucí. Jejich velký počet komplikuje vyhodnocení výsledků, protože některé jsou anomální a jiné ne. Je také nutné uvést, že úhly nejsou homogenní a jejich velikosti a změny mezi jednotlivými snímky se podstatně liší. Rozsah pohybu hlavy je například daleko menší než rozsah pohybu ruky. To ovlivňuje i hodnotu rekonstrukční chyby autoenkodéru, popsaného v kapitole 3.2
Nakonec byla vybrána následující sérii úhlů, s cílem sledovat anatomické vlastnosti těla řidiče. Pro validaci vhodnosti vybraných úhlů je v kapitole 6. provedeno experimentální srovnání s jinými konfiguracemi.

Byla vybrána následující série úhlů:

1. Předloktí levé ruky – Body #2 #3 #4
2. Předloktí pravé ruky – Body #5 #6 #7
3. Levé rameno – Body #3 #2 #1
4. Pravé rameno – Body #1 #5 #6
5. Osa mezi očima a nosem – Body #15 #0 #16
6. Osa mezi ušima a nosem – Body #17 #0 #18
7. Osa mezi pravým uchem nosem a okem – Body #17 #0 #15
8. Osa mezi levým uchem nosem a okem – Body #18 #0 #16
9. Osa ramen – Body #2 #1 #5
10. Náklon krku do levé strany – Body #2 #1 #0
11. Náklon krku do pravé strany – Body #5 #1 #0



Obrázek 4.3: Sada příznaků základního navrhovaného řešení

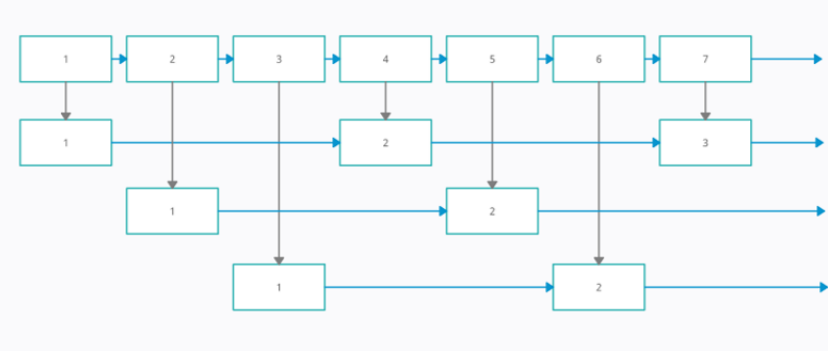
Základní konfigurace příznaků, tedy obsahuje pouze 12 klíčových bodů, které dále tvoří 11 úhlů. Dimenzionalita dat je tedy snížena o více než 50%. Úhly respektují anatomické závislosti lidského těla. Cílem výběru je pokrytí všech kloubů, končetin, krku a několika závislostí na hlavě. Předpokladem je, že neuronová síť následně vybere ty nejdůležitější z nich. Výstupní vektor úhlů je vytvořen pro každý jednotlivý snímek a celé video pak slouží jako vstup pro algoritmus detekující anomálie. Takto vytvořené vektory příznaků je třeba rozdělit do vstupních dávek, aby je bylo možné použít jako tréninková a testovací data.

4.2 Formát dat a vstupních dávek

Protože jednotlivé konfigurace příznaků mohou být v jistých situacích normální a v jiných anomální, je vhodné vyhodnocovat vstupy v podobě dávky. To snižuje šanci na shodu a také přináší dodatečnou informaci v podobě časové závislostí jednotlivých snímků. Následuje popis, jak jsou vstupní dávky konstruovány, a dále vysvětlení, jak jsou zjištěné závislosti zohledněny v datech a co vlastně znamenají.

V případě tréninkového i testovacího videa každý vstupní vektor příznaků zachycuje jeden snímek. Frekvence snímků je 30fps. Pokud je cílem zachytit pohyb v čase, je dobré dělat mezi snímky mezery. Několikamilimetrové rozdíly v pohybu mezi dvěma snímky totiž nejsou dostatečné pro vyhodnocení anomálnosti pohybu. V případě například mezery délky 3 bude počet snímků za vteřinu videa 10.

Dávka 16 příznaků tak znamená 1,6 sekundy ve videu. To by se mohlo zdát jako plýtvání, protože to znamená využití pouze 1/3 video záznamu, naštěstí existují metody, které tento problém řeší. Pokud je použita sekvence velikosti 3, je možné rozdělit video na tři částí. Každá část bude posunutá vůči originálu o daný počet obrazových snímků. Do které části bude snímek patřit se dá vyhodnotit pomocí funkce modulo, jak je zobrazeno níže. Tímto se dá z jednoho videa dostat větší počet vstupních sekvencí, kdy pro první sekvenci budou přidány snímky č. mod 3=0.



Obrázek 4.4: Konstrukce vstupních sekvencí

Velikost mezery mezi snímky a délka/formát vstupní sekvence, jsou velice důležité především s ohledem na použitou architekturu neuronové sítě. Lze je chápat jako další hyperparametry, které neuronovou síť ovlivňují poměrně zásadním způsobem. Délka vstupní sekvence je jedním ze základních parametrů použité rekurentní sítě. Určuje, jak velká série příznaků bude brána v potaz u detekce anomálie. Jak bylo uvedeno výše, mezera mezi snímky určuje, jak daleko jsou jednotlivé snímky od sebe a navíc určuje, kolik bude mít každá série příznaků snímků. V kombinaci pak definují formát vstupní sekvence. Jejich vzájemné nastavení ovlivňuje výsledné vlastnosti detekce následujícím způsobem:

1. Pokud je velikost vstupní dávky 1, neexistují časové závislosti a neuronová síť rozpoznává pouze závislosti mezi příznaky jednoho snímku.
2. Se vzrůstající velikostí dávky se zvyšuje počet snímků, které se vyhodnocují najednou. To pomáhá zvýšit kvalitu detekce anomálií, ale zároveň se snižuje schopnost sítě rozeznat normální data, protože je těžší zjistit, kam povede každý následující pohyb. Důsledkem je pak větší šum na výstupu dekodéru u validační sady. Je také těžší určit, kde přesně začíná a končí hranice anomálie.
3. Se vzrůstající velkostí mezery se rovněž zvyšuje obtížnost předpovědi pohybu. Zvyšuje se však i jeho rychlost, tedy změna polohy mezi snímky.
4. Delší délka vstupní dávky po případě délka mezery znamená zpoždění předpovědi o daný počet snímků, což může znamenat poměrně velký problém a je nutné přihlédnout k obecně uznávaným kritériím, jako je např. reakční čas řidiče odhadovaný na dvě vteřiny.

Po poměrně rozsáhlé experimentaci byla vybrána velikost dávky 8 s mezerou mod 3 snímky. Vyšší hodnoty již nijak nepomáhaly u detekce. Vždy je vyhodnoceno cca 10 snímků za vteřinu, což znamená, že je analyzována zhruba jedna sekunda videa. Tato hodnota je stále předpověditelná a zároveň je stále možné jednoznačně určit, kde začíná a končí anomálie. Čas pod 2 sekundy se také zdá být poměrně bezpečným zpožděním detekce. Vstupní dimenze problému je tedy 11 x 8. Takto upravené vstupní dávky se již dají použít pro trénink neuronové sítě. Nejdříve je nutné si tuto síť popsat.

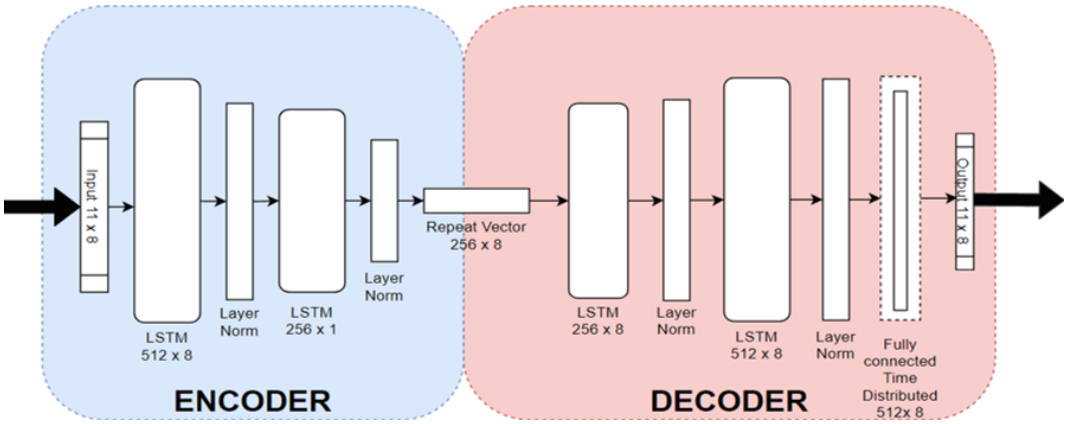
4.3 Architektura základního řešení

Tato kapitola se zabývá architekturou autoenkodéru, použitého v navrhovaném základním řešení. Budu se věnovat především vysvětlení nastavení hyperparametrů, ale pouze těch, které jsou pro toto řešení takřka univerzální a změna jejich nastavení by byla buďto fatální pro celkovou efektivitu sítě - nebo naopak naprosto nijak neovlivňující výsledky. Zbylé nastavení parametrů je provedeno experimentálně v kapitole 5.

Autoenkodér má poměrně jasně definované vlastnosti a pevně dané prvky. Výhody plynoucí z využití časových závislostí mezi daty pro detekci anomálie, byly již zmíněny v předchozí kapitole. Detekce se dá provést s využitím jak rekurentních, tak standardních vrstev neuronové sítě. V případě standardních, (tzv. fully-conected) sítí se v praxi využívá vstupního vektoru nepronikajících oken a naprosto ignoruje časové závislosti mimo vektor. Hojně se využívá také přístup pohyblivého okna, který je však poměrně problematický, protože neuronové síti definuje jasně délku závislostí, kterou v datech detekuje.

Proto jsou v základním řešení využity především jednosměrné LSTM vrstvy. LSTM vrstvy nejen samy rozhodují, kterou část vstupního vektoru budou brát v potaz, ale zároveň dovolují vytvoření závislostí mezi dávkami samotnými. Během návrhu tohoto řešení bylo zjištěno, že je výhodné předat poslední skrytý stav buňky pro každou sekvenci tréninku a využít tzv. statefull LSTM, při současném použití aktivační funkce *tanh*, místo aktivační funkce *relu*. Důvodem je, že při použití statefull LSTM dochází k explozi gradientu u druhé zmíněné možnosti. Tento jev se sice povedlo redukovat také pomocí tzv. gradient clippingu, ale výsledky byly mírně horší než při použití funkce *tanh*.

U autoenkodéru je jedním ze zásadních parametrů "tvar" vrstev – tedy jejich snižující se velikost směrem k úzkému hrdlu a následné rozšíření směrem od něj. Počet těchto vrstev a jednotek v nich je však závislý na vstupní datové sadě. Během návrhu bylo experimentálně zjištěno, že nejvhodnější počet vrstev enkodéru/dekodéru je mezi dvěma a čtyřmi vrstvami. Stačí to na reprezentaci vstupních příznaků. V případě vyššího počtu vrstev se vlastnosti sítě prudce zhoršují. V navrhovaném základním řešení je využita pouze jedna fully-connected vrstva před výstupem.



Obrázek 4.5: Architektura základního navrhovaného řešení

Vliv počtu jednotek vrstvy na celkovou efektivitu je pak daleko jemnější a jejich ideální počet je ovlivněn následujícími hyper-parametry.

- Počet příznaků.
- Dimenze příznaků.
- Počet časových kroků, které má síť vyhodnotit.
- Frekvence kroků vzhledem k obrazovým snímkům.

Počet jednotek je pro vstupní sekvenci popsanou v předchozí kapitole efektivní do 512ti jednotek pro první, největší vrstvu. Po poměrně rozsáhlé experimentaci, popsané v kapitole 5 se jeví jako nejlepší varianta použití dvou vrstev v každé z částí. Pro první bylo použito 512 jednotek a pro druhou 256.
Téměř všechny LSTM vrstvy předávají následujícím vrstvám celou sekvenci skrytých stavů, s výjimkou poslední vrstvy enkodéru. Aby zde došlo ke kompresi, je vrácen pouze poslední skrytý stav, který je následně opakován a kopírován tzv. repeat-vectorem. To je provedeno, protože je nutné zachovat dimenzionalitu výstupu enkodéru a předat jej tak dekodéru.
Jako nejvhodnější se jeví v tomto případě použití normalizace vrstev skrze osu *#1* po každé LSTM vrstvě. I když je tento postup poněkud netradiční, s jeho pomocí bylo dosaženo opakovaně nejlepších výsledků pro tuto architekturu sítě. Pro inicializaci vah byla využita standardní Xavierova inicializace, popsaná v kapitole 3.4.1. Takto navržená neuronová síť byla trénována postupem popsaným níže.

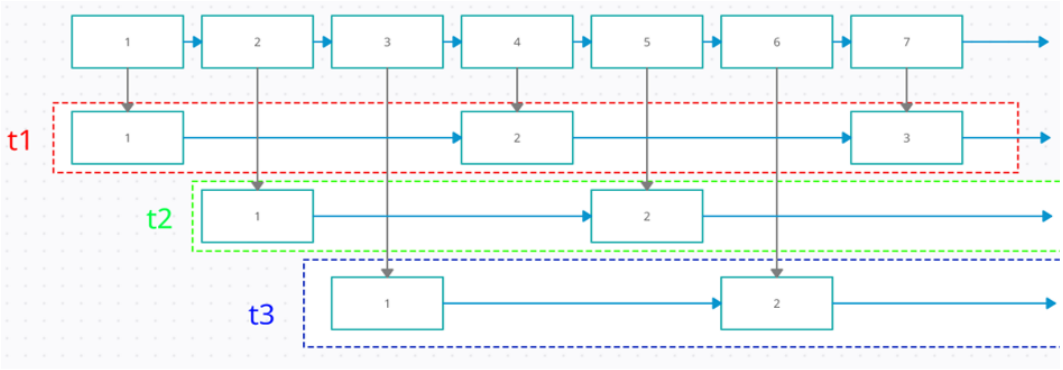
4.4 Návrh tréninkového procesu

Tato kapitola popisuje základní prvky tréninkového procesu použitého při návrhu základního řešení. Během tréningu je neuronové sítí předávána sada neobsahující anomálie. Opět je nutno počítat s několika hyperparametry, jejichž nastavení ovlivňuje efektivitu učícího procesu. Krom počtu

epoch a batch-size – počtu sad příznaků, po kterých se provede update gradientu chyby, jde samozřejmě i o optimalizační algoritmus samotný a jeho alpha a beta parametry (učící konstantu nebo také learning rate a momentum).
Nejspíše nejdůležitějším hyperparametrem, je učící konstanta, která je přímo závislá na použitém optimalizačním algoritmu. Během implementace byly testovány algoritmy SGD [40] a Adam [39]. V případě SGD bylo nastavení a efekt velice nejisté, navíc SGD jako takový se v praxi v podstatě nepoužívá. Adam nabízí navíc možnost využití tzv. momentum principu, který zjemní update standardního gradientu s využitím dávek. Dojde tak ve většině případů k zrychlení optimalizačního procesu a hledání minima. Pro standardní LSTM nebyl důvod jakkoli měnit učící konstantu, learning rate. Bohužel po změně na statefull LSTM docházelo během optimalizace k uváznutí na lokálním minimu. Tento problém byl vyřešen snížením learning-rate na 10^{-4}. Nastavení beta parametru bylo ponecháno na algoritmu Tensorflow.
Trénink po tomto nastavení probíhal poměrně bez potíží a k explozi gradientu za použití *tanh* funkce nedocházelo. LSTM buňky jsou poměrně citlivé na správnou velikost tréninkové dávky tzv. batch-size. Nastavení této hodnoty na 8 zaručilo schopnost algoritmu konvergovat k optimálním hodnotám za zhruba 25 epoch. Vyšší počet epoch nemá žádný viditelný přínos pro výsledky tréninku. Naopak nižší počet epoch pro navrženou architekturu způsoboval neschopnost rekonstruovat během inference výstupy. Proces inference je poměrně přímočarý, nicméně je dobré se mu alespoň v krátkosti věnovat.

4.5 Proces inference a vyhodnocení chyby

Během inference je neuronové síti předávána validační sada obsahující anomálie. Neuronová síť tyto data dostává sekvenčně. Proces inference v čase funguje podobně jako klasický princip pohyblivého okna o pevné šířce. Neuronová síť vždy zpracovává 8 snímků s mezerou mod 3. Jak bylo zmíněno v kapitole 4.2, výstup je pak opožděn.



Obrázek 4.6: Princip pohyblivého okna u inference

Výstupem není přímá předpověď anomálnosti dat, ale pouze rekonstruovaná reprezentace vstupu. Pokud je tato reprezentace kvalitní, dá se předpokládat, že se neuronová síť s podobnými daty setkala během tréninkového procesu, a proto jsou tato data považována za normální. Pro analýzu normálnosti dat je potřeba zvolit vhodnou chybovou funkci. Pro potřeby výpočtu byla použita funkce střední kvadratické chyby, která je definovaná rovnicí níže.

$$MSE(\hat{\theta}) \stackrel{def}{=} [(\hat{\theta} - \theta)^2] \tag{4.2}$$

Nicméně takto vypočtenou hodnotu stále nelze považovat za výsledek. Je nutné ji vyhodnotit dalším algoritmem, což je v tomto případě ROC křivka popsaná v kap. 3.5 a křivka přesnosti a citlivost, která nabízí jiný pohled na schopnosti sítě. Tato práce se nezabývá hledáním perfektního řešení, jako daleko důležitější se jeví hodnota AUC těchto charakteristik. Cílem rovněž není hledání optimální hodnoty hranice chyby, kterou by bylo možné určit pouze s pomocí daleko obsáhlejší datové sady. Důležitým krokem je však validace funkčnosti řešení popsaného v této kapitole. Následující kapitola se zabývá experimentálním ověřením funkcionalit navrženého řešení.

Kapitola 5

Experimentální realizace navrhovaného základního řešení

Tato kapitola popisuje experimentální realizaci základního řešení. Cílem je dokázat, že tento postup vede k detekci anomálií rozeznatelných zvýšenou rekonstrukční chybou. Zároveň je v tomto experimentu ověřena vhodnost použitých obrazových příznaků a schopnost generalizace pro různé automobily a řidiče, a to pomocí srovnání se stimulátorovými videozáznamy. Základem je řešení podrobně popsané v kapitole 4. Úkolem experimentální realizace není odvození všech jeho hyperparametrů. Ověřuje však některé závěry, které byly v kapitole učiněny. Závěr kapitoly je věnován jisté polemice o vhodnosti některých navržených prvků a jejich možném zlepšení.

Kapitola je členěna následujícím způsobem:

- **Kapitola 5.1** obsahuje metodiku, jak připravit data do požadovaného formátu, jak postupovat při tréninku, jak vybrat vhodné řešení a následně provést vyhodnocení.
- **Kapitola 5.2** obsahuje vyhodnocení výsledků dosažených v průběhu experimentální realizace navrhovaného základního řešení
- **Kapitola 5.3** je věnována polemice, diskusi a závěrům experimentu a možným vylepšením.

5.1 Popis metodiky základního řešeni

Tato kapitola je opět rozdělena na několik částí, popisujících metodiku jednotlivých kroků experimentu. Je zde hlouběji popsána datová sada, její příprava a předpoklady. Jedná se o videa poskytnutá pro tuto práci školou a také o videa, která byla pořízena pomocí simulátoru vyrobeného přímo pro potřeby této práce.
Následně jsou popsány jednotlivé kroky provedeného experimentu a vyhodnocení. Je nutné poznamenat, že jak v případě tohoto řešení, tak v případě jednotlivých experimentů, předcházel realizaci

a vyhodnocení proces několikadenního iterativního prohledávání batch-size, velikosti vstupní dávky, počtu vrstev, jejich hloubky a dalších hyperparametrů. Tímto byly alespoň částečně ověřeny globální vlastnosti navrhnuté implementace. Navrhnuté řešení proto nelze považovat za nahodilé. Výsledné řešení je nejstabilnější a při srovnání s ostatními před-experimentálními variantami také nejvýkonnější.

5.1.1 Příprava datové sady

Datová sada se skládá ze dvou částí - videí, zaznamenanými se třemi řidiči během reálného řízení vozu a videí které byla pořízena v simulačním prostředí. To se skládá z volantu a sedačky přímo určených pro simulaci vozidel, a dále kamery připevněné na stativu v podobném umístění, jaké bylo použito při pořizování zbylých videí. V takto sestaveném prostředí bylo pořízeno několik videí, aby bylo možné porovnat lépe vlastnosti navrženého systému. Videa se dají rozdělit na tréninková, obsahující pouze normální chování a validační obsahující i anomální.
U tréninkových videí bylo do poskytnuté datové sady přidáno další video h2_norm_flip. Toto video je kratší než zbytek videí. Záměrně obsahuje netypickou figurantku, která svým chováním a proporcemi není podobná ostatním řidičům. Video je také natočeno z mírně jiného úhlu a jinou kamerou, jak by tomu bylo v jiném automobilu. Cílem takto obohacené datové sady je srovnat generalizační schopnosti systému.

Obrázek 5.1: Figurantka simulátorového videa

Tréninková video sada obsahuje videa ze dvou zdrojů:

- **h2_norm_flip** – stimulátorové video zaměřené na test chování různých datových sad.
- **normal_china_face_body_geordi_small, normal_china_face_michael_small, normal_china_face_body_rados** – hlavní část tréninkových videí.

Validační videa jsou opět natočena nejen za reálného běhu, ale také s pomocí simulátoru. Videa bylo nutné nejdříve anotovat a označit v nich alespoň orientační začátek a konec anomálie. Naprosto přesné určení není možné, nicméně se dá odhadnout, kdy ke stavu došlo, a to s přesností několika stovek snímků.

Validační sada obsahuje tyto videa:

- **anom-1, anom-2, anom-3** která jsou zaznamenána za běžného řízení vozu.
- **anom-4** natočené na simulátoru s řidičem viděným během tréningu.
- **anom-5** natočené na simulátoru s řidičem, který není přítomen v tréninkové sadě.

Z těchto videí byla pomocí Open-Pose extrahována poměrně velká série obrazových příznaků, v tomto případě úhlů v rozmezí mezi 0°-180°. Jde o sadu různých možných úhlů, ze které využívám pouze sadu příznaků respektující anatomické souvislosti řidiče, definované v části 4.1.2. Tento postup dovoluje využívat různé konfigurace příznaků pro různé experimenty, bez nutnosti převodu videa pro každou konfiguraci zvláště. Z každého video záznamu je po převodu vrácen jeden vektor příznaků, tedy dávka videa o délce d $\times$ 11. Takto extrahovaný příznakový vektor je třeba převést na vstupní dávky o velikosti n $\times$ 8 $\times$ 8 $\times$ 11.

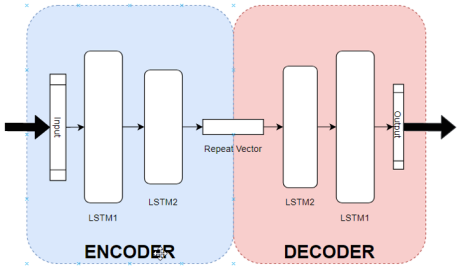
- Aby LSTM pracovalo vždy s příznaky pocházejícími ze stejného záznamu, je nutné každou dávku videa upravit tak, aby délka vektoru příznaků byla dělitelná délkou vstupní sekvence LSTM. Toto musí vyhovovat vztahu mod 8 = 0, zbylé snímky jsou odstraněny.
- Z jednoho videozáznamu je takto vyextrahováno několik dávek videa, jejichž počet je dán mezerou mezi snímky (pro tento případ jsou to 3 sady), jak je popsáno v kapitole 4.1.2.
- Sady jsou následně spojeny do vektoru videa, pro jehož délku platí stejné podmínky vzhledem k batch-size. Délka vektoru musí být také rovna mod 8 = 0.
- Takto sestavené sady příznaků jsou následně spojeny - tentokrát do finálního vektoru, který je vstupem pro neuronovou síť. Tato úprava byla provedena jak pro tréninková, tak validační videa.

5.1.2 Experimentální hledání optimální architektury

Jak je popsáno v kapitole 4.2, byla zvolena architektura založena téměř čistě na statefull LSTM vracejícím sekvenci skrytých stavů. Přechod na statefull LSTM zvedl globálně úspěšnost klasifikace o 2-5 %. Je tomu tak nejspíše, protože nezohledňuje pouze závislosti skrytých stavů uvnitř dávky

(batch-size), ale i mimo ni. Každá LSTM vrstva je následována vrstvou normalizace. Bylo provedeno několik malých experimentů se změnou jejího umístění po případě úplného odstranění, nicméně to nevedlo ke zlepšení. V základním řešení nebylo použito žádné fully-conected vrstvy, krom poslední, sloužící ke změně výstupní dimenze.
Jak bylo uvedeno v kapitole 4.3, optimální počet vrstev pro maximální kvalitu klasifikace je mezi 4 a 2. Byly využity pouze dvě vrstvy pro enkodér a dekodér, což vede k relativně malému počtu parametrů. Velikost jednotlivých vrstev neuronové sítě je určena experimentálně - srovnáním několika konfigurací. První hodnota je vždy nastavením první a poslední vrstvy a hodnota druhá je vždy nastavením vrstvy umístěné uprostřed architektury.

- 512-256
- 128-64
- 64-32
- 32-16



Obrázek 5.2: Základní architektura řešení

Tréninkový proces je prováděn s dávkami o velikosti 8 snímků a mezerou mod 3, jak je popsáno v kapitole 4.2. Dávky jsou nepronikající. Zvažovanou součástí experimentu bylo použití principu pohyblivého okna pronikajících dávek, jak je tomu u inference. Výsledky byly zhruba stejné a doba tréningu se prodloužila. Tréninkový proces je prováděn v 25ti epochách, kdy bylo zjištěno, že u menšího počtu docházelo ke snížení stability tréningu a u nastavení vyššího k žádnému zlepšení nedošlo.
U inference byla nejprve provedena změna formátu validačních dat tak, aby na nich bylo možné simulovat její provedení v reálném čase v souladu s principem pohyblivého okna pevné délky. Rekonstruovaný výstup je porovnaný se vstupem pomocí funkce střední kvadratické chyby. Oba principy jsou popsané hlouběji v kapitole 4.4.1. Výsledky jsou uloženy do chybového vektoru, který je pro každou konfiguraci následně vyhodnocen způsobem popsaným v další kapitole.

5.1.3 Způsob vyhodnocení výsledků

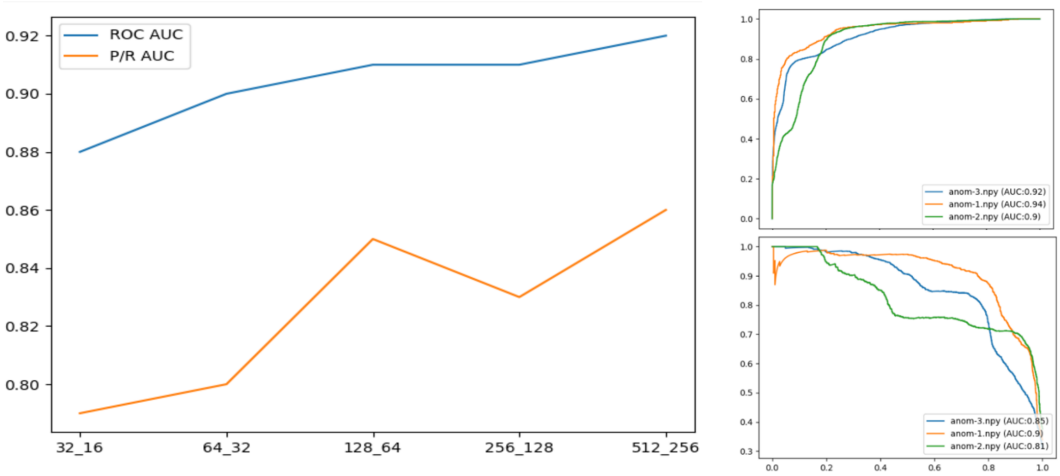
Pro zmírnění vlivu náhodnosti je vyhodnocení provedeno s různými nastaveními neuronové sítě a různými, pevně danými nastaveními náhodného generátoru čísel. Při hledání optimální architektury byl proces tréningu a inference zpuštěn třikrát pro každou konfiguraci s různými nastaveními náhodného generátoru.
Nejprve je pomocí validačních sad - anom-1, anom-2 a anom-3 provedeno hledání optimální architektury. Následně jsou vyhodnoceny tři nastavení generátoru náhodných čísel pro každou architekturu vybrané velikosti. Pro každou velikost je vypočtena průměrná hodnota z průměrného výsledku je-

jich validačních sad charakteristik ROC AUC a P/R AUC. Tyto hodnoty jsou následně srovnány mezi sebou.

Pro konfiguraci s nejlepšími celkovými výsledky jsou nakonec srovnány výsledky jednotlivých sad. U anom-3 je pak také analyzována hladina chyby, za účelem nalezení souvislostí v schopnosti detekovat určité anomálie. Součástí experimentu je porovnání vlastnosti videí natočených v automobilu a na simulátoru pomocí datové sady anom-4. Kde je cílem srovnat možnosti detekce v jiném prostředí s minimální datovou sadou. Následně s pomocí datové sady anom-5, obsahující řidiče neviděného v žádné tréninkové sadě, je vyhodnocena schopnost generalizace.

5.2 Hodnocení výsledků experimentální realizace základního řešení

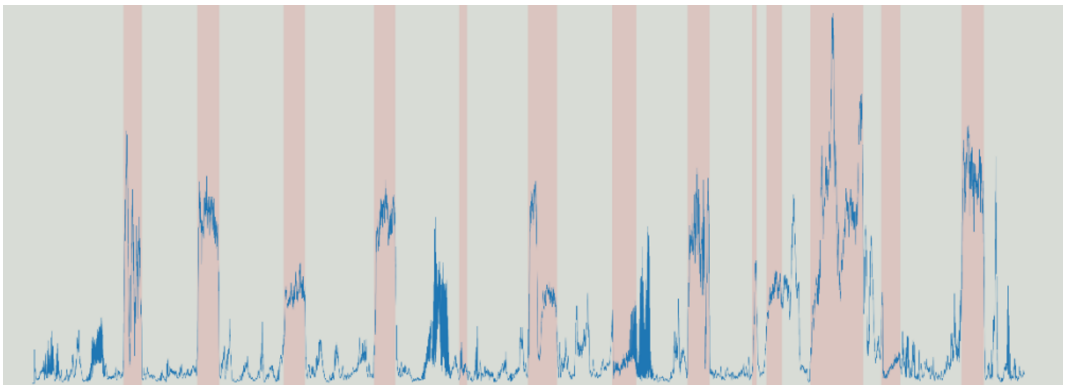
Nejdříve byl proveden test vyhledávání optimální architektury popsaný v předchozí kapitole. Výsledky byly dosti podobné a lišily se pouze v rozmezí 4% pro ROC AUC a 7 % pro P/R AUC. Pokud je autoenkodér trénován správným způsobem a využívá dobře připravenou datovou sadu, výsledky se zásadním způsobem neliší. Nejlepších výsledků dosáhla konfigurace 512/256. Na obrázku níže je vidět průměrné výsledky zaznamenané za tři různé běhy pro jednotlivé architektury a výsledky té nejlepší.



Obrázek 5.3: Grafy hodnot ROC AUC a P/R základního navrhovaného řešení

U pohledu na výsledky nejlepšího běhu jsou viditelné poměrně velké skoky u ROC a P/R křivek. Důvodem tohoto chování je nedokonalá detekce některých anomálií. To je způsobeno nedostatečnou hodnotou chyby u některých anomálií, a tedy složitější separaci anomálních a normálních dat pro některé situace.

Na chybovém grafu (obr. 5.4) je vidět, že absolutní chyba je dobře odhadnuta pro všechny anomálie kromě #5 a #12, nedokonalá detekce anomálií #6 #7 #9 #10 a slabá detekce anomálie #3. To koreluje se situacemi simulujícími mikrospánek na videu, jak je vidět v tabulce níže.



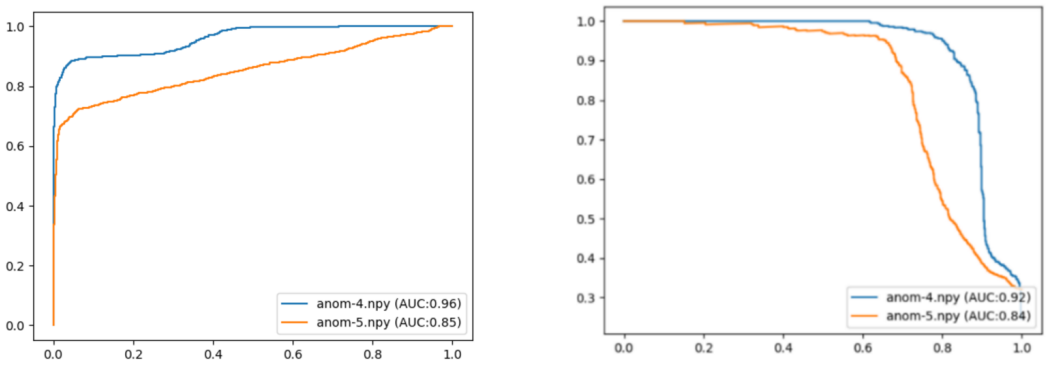
Obrázek 5.4: Graf znázorňující MSE chybu pro sadu anom-3 základního navrhovaného řešení

#	Start anomálie a její konec	*Typ anomálie*
1	1167-1406	*kašel*
2	2138-2402	*kašel*
3	3156-3532	*spánek – hlava dozadu*
4	4416-4701	*kašel*
5	5367-5631	*únava – hlava dol*ů
6	6391-6798	*únava – hlava dozadu*
7	7442-7826	*únava – hlava dol*ů
8	8491-8764	*kašel*
9	9314-9379	*únava*
10	9481-9715	*spánek*
11	10082-10762	*telefonovaní*
12	10950-11249	*únava – hlava do boku*
13	12035-12326	*kašel*

Tabulka 5.1: Anotace anomálií videa anom-3

V situacích během simulovaného řízení vozu (reprezentovaných anom-3) bude dostačující i slabá nebo velmi slabá detekce. Figuranti samozřejmě pouze naznačovali jednotlivé situace, aby bylo řízení naprosto bezpečné. U zadání bylo vedoucím práce doporučeno podobnou simulaci provést raději lokálně v simulačním prostředí. Prostřednictvím validačních sad anom-4 a anom-5 je vyhodnocena schopnost generalizace základního řešení. Protože, byla videa nahraná na simulátoru jde o realistické, výraznější a lépe detekovatelné reakce, než je možné si dovolit na videu natočeném na parkovišti, aniž by došlo k ohrožení ostatních účastníků silničního provozu.

Jak jde vidět na obrázku 5.5, model je schopen generalizovat i pro anom-4, které je reprezentováno ve tréninkové sadě pouze jedním poměrně krátkým videem. To znamená, že model dokáže kombinovat znalosti pocházející z několika videí. Video anom-5 je hráno hercem (autorem práce), který není zahrnut v tréninkové sadě, a i přes snížení kvality detekce je zřejmé, že model dokáže pracovat i s naprosto neznámými řidiči.



Obrázek 5.5: Grafy ROC a PR křivky pro validační sady anom-4 a anom-5 základního navrhovaného řešení

5.3 Diskuse

Přípravě tohoto experimentu předcházela dlouhá série menších, více či méně strukturovaných experimentů, a řešení tedy rozhodně není nahodilé - spíše je evolucí předchozích pokusů. Původní model byl pouze jednoduchou LSTM sítí. Kdy jednoduché LSTM detekovalo některé chyby zřetelněji, nicméně výsledky v podobě ROC AUC a P/R AUC hodnot byly o 3-5% horší než tomu je u navrhovaného řešení. Experimentace s různými formáty vstupní dávky, její délky nebo počtu vynechaných snímků nedosahovala lepších výsledků. Je tomu tak nejspíše, protože systém nedokázal přesně určit hranice anomálie a problematické anomálie, uvedené v tabulce 1.1 zůstávaly špatně rozpoznatelné.

Jak bylo během experimentů zjištěno, je velice těžké porovnat anomálie v oblasti hlavy řidiče. Jednou z příčin může být to, že se většina příznaků v oblasti hlavy pohybuje současně, tedy stejným směrem. Je nutné srovnat jejich anomálnost pomocí náklonu, jako je tomu u osy očí nebo pozice očí vůči ramenům. To může být problematické, protože k podobným situacím dochází i u validačních videí, kdy například řidič pouze pohlédne směrem dolů. Proto hodnota rekonstrukční chyby není tak velká a detekce anomálie není tak snadná. Počet těchto příznaků v porovnání s celkovým počtem je také poměrně malý, což způsobuje že, ostatní příznaky skupinově potlačí vliv anomální pozice jiného. Dalším prvkem, který komplikuje detekci, je velikost změny úhlů. Protože je poměrně malá, z ní plynoucí velikost rekonstrukční chyby také není velká. Její výsledný vliv na pozitivní (anomální) klasifikaci proto není často dostačující.

Teoreticky, by se dal tento jev zmírnit vhodnou volbou jiných příznaků. Následuje tedy několik experimentů, zaměřených na použití alternativních konfigurací obrazových příznaků. Prvním typem jsou klíčové body bez konstrukce úhlů. Druhým typem příznaků je série úhlů zaměřených především na oblast hlavy a její pozici vzhledem ke zbytku těla. Je předpokládáno, že anomálie popsané v předchozím odstavci jsou detekovatelné především změnou pozice hlavy a síla detekce zbylých anomálií je dostatečná na to, aby byly detekovány menším počtem příznaků.

Kapitola 6

Alternativní reprezentace obrazových příznaků

Tato kapitola se věnuje alternativní reprezentaci obrazových příznaků a jejím srovnáním se základním navrhovaným řešením v kapitole 5. Tímto jsou myšleny vybrané úhly konstruované pomocí klíčových bodů. Cílem je popsat navrhované varianty před jejich experimentálním srovnáním v další kapitole. Pro srovnání jsou využity přímo klíčové body a také série úhlů, které jsou vybrány s cílem, zlepšení detekce problematických anomálií.

Kapitola je strukturována následujícím způsobem:

- **Kapitola 6.1** se zabývá popisem příznaků vytvořených pouze klíčovými body.
- **Kapitola 6.2** se zabývá popisem příznaků vytvořených sadou úhlů detekujících především změny v pozici hlavy.

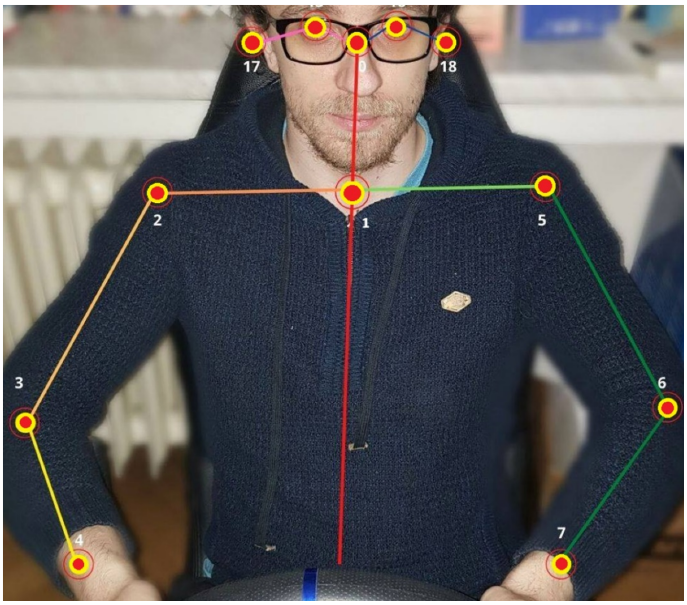
6.1 Příznaky tvořené klíčovými body

Příznaky tvořené klíčovými body jsou representovány souřadnicemi x a y. Pomocí Open-Pose jsou tyto hodnoty během extrakce normalizovány do rozmezí mezi -1 a 1. Toto je vhodné pro zlepšení vlastností tréninku a pro zmenšení vlivu formátu videa nebo různých řidičů. Výstupem je pole souřadnic o rozměrech 12×2 pro každý snímek videa. Je nutné ho transformovat pro neuronovou síť do formátu 24×1, kdy souřadnice x a y jsou v poli hned za sebou.
Příznaky jsou vybrány opět na obou stranách těla, i když se může zdát, že je to nadbytečné. Například ramena nebo oči mohou detekovat anomálii efektivněji na jedné straně než druhé. To je způsobeno především pozicí kamery, jak je ukázáno na obrázku 6.1.

Vybrané body splňují následující účel:

- Zápěstí pro levou ruku – Bod #4
- Zápěstí pro pravou ruku – Bod #7
- Loket pro levou – Bod #3
- Loket pro pravou ruku – Bod #6
- Rameno pro levou ruku – Bod #2
- Rameno pro pravou ruku – Bod #5
- Krk – Bod #1
- Levé ucho – Bod #17
- Pravé ucho – Bod #18
- Nos – Bod #0
- Levé oko – Bod #15
- Pravé oko – Bod #16

Použití klíčových bodů má několik nepříznivých vlastností. Dva rozlišní řidiči budou mít naprosto rozdílné pozice bodů pro stejné nebo podobné situace. Úhly přidávají informaci o závislosti, kterou by měl systém sledovat. Klíčové body tuto výhodu nemají - zjištění triviálních závislostí, například že hlava je nad úrovní rukou, je tedy čistě na neuronové síti. Ta pak potřebuje více dat, aby dokázala tento problém vyřešit, a to i v rámci jednoho řidiče. Z toho plyne i značná složitost přenosu znalostí mezi několika řidiči.



Obrázek 6.1: Alternativní obrazové příznaky - Klíčové body

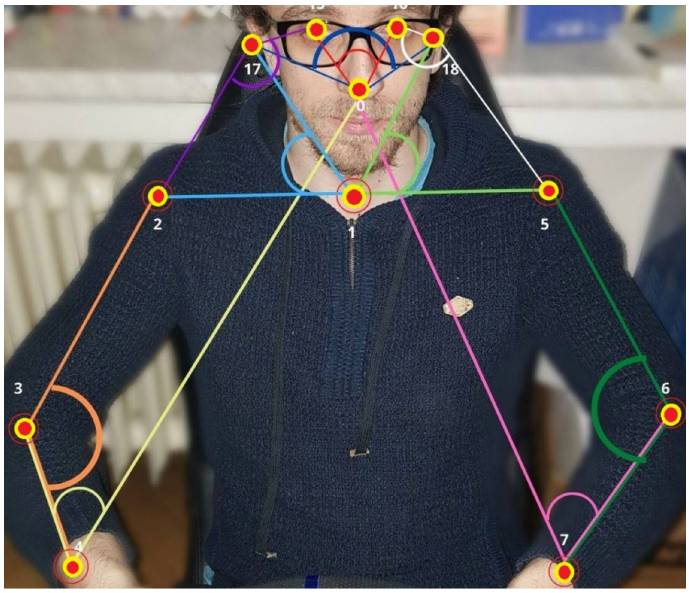
6.2 Selekce úhlů zaměřených na těžko detekovatelné pózy

Při bližší analýze základního navrhovaného řešení jsou vidět některé těžko detekovatelné pózy. Tyto pózy souvisí bezezbytku s mikro-spánkovými situacemi. Jedním z možných řešení je změna úhlů, tzn. zvolit takové, které budou lépe detekovat tyto situace. Vybrané úhly jsou zaměřeny především na oblast ramen a hlavy, protože v těchto místech docházelo k problematickým pohybům. Patří mezi ně například pokles hlavy nebo změna osy ramen. Jednotlivé úhly byly vybírány z větší množiny. Tato byla následně iterativním prohledáváním za pomoci jednoduchého autoenkodéru porovnána. To sice neříká nic o jejich detekčních vlastnostech, protože tím nejsou nijak zohledněny jejich vzájemné vztahy. Na druhou stranu tento postup výrazně snižuje pravděpodobnost, že bude vybrán zcela nevhodný úhel.

Jde o následující sérii úhlů:

- Osa mezi očima a nosem – Body #15 #0 #16
- Loket levé ruky – Body #2 #3 #4
- Loket pravé ruky – Body #5 #6 #7
- Osa mezi ušima a nosem Body – #17 #0 #18
- Osa mezi krkem a pravým uchem – Body #2 #1 #17
- Osa mezi krkem a levým uchem – Body #5 #1 #18
- Osa mezi levým okem uchem a ramenem – Body #15 #17 #2
- Osa mezi pravým okem uchem a ramenem – Body #16 #18 #5
- Osa mezi levým předloktím a nosem – Body #3 #4 #0
- Osa mezi pravým předloktím a nosem – Body #6 #7 #0

Počet úhlů je 11, výstupní vektor tedy bude 11×1. Jde stále o malou množinu nicméně je možné, že neuronová síť základního navrhovaného řešení pro ni nebude optimální, protože jde o reprezentaci jiných závislosti mezi klíčovými body.
Některé úhly vychází ze stavby těla, jako tomu bylo v základním řešení. Byly přidány také úhly, které jsou založeny na očekávané póze řidiče u řízení vozu. Patří zde například osa mezi levým předloktím a nosem nebo její zrcadlová varianta na pravé straně. Pokud bude řidič řídit a situace je standardní, ruce položené na volantu mají určitý uhel vzhledem k hlavě. V případě anomálie a pádu horní poloviny těla do jakéhokoli směru se bude dynamicky měnit, stejně jako v případě změny pozice ruky vůči volantu.



Obrázek 6.2: Alternativní obrazové příznaky - Úhly vycházející z pozice hlavy

Následuje experimentální srovnání těchto sérií příznaků, a dále popis, jakým způsobem se chovají v součinnosti s navrženým řešením.

Kapitola 7

Experimentální realizace použití alternativních obrazových příznaků

Tato kapitola se zabývá popisem experimentální realizace využití alternativních příznaků pro detekci anomálií, popsanou v kapitole 2. Je provedeno srovnání dvou konfigurací. Po krátkém popisu metodiky následuje samostatné srovnání a vyhodnocení každé z variant vzhledem k základnímu řešení. Diskuse je pak vedena především v návaznosti na diskusi v kapitole 5.3.

Kapitola je členěna následujícím způsobem:

- **Kapitola 7.1** popisuje metodiku jednotlivých experimentů.
- **Kapitola 7.2** vyhodnocuje výsledky experimentů.
- **Kapitola 7.3** je věnována diskusi vyplývající z výsledků.

7.1 Popis metodiky experimentální realizace řešení s využitím alternativních příznaků

Tato kapitola se zabývá popisem metodiky navrhovaného řešení. Metodika vychází z metodiky, která již byla popsána v kap. 5.1. Metodika pro přípravu datové sady v podstatě popisuje pouze specifika nutná pro převod obrazových příznaků. V rámci kapitoly jsou rovněž řešeny změny v konfiguraci neuronové sítě. Dále jsou popsány důvody, které k těmto změnám vedly. Kapitolu uzavírá způsob vyhodnocení výsledků (obdobně jako kap. 5.1.3).

7.1.1 Příprava datové sady

Datová sada opět sestává ze dvou částí - normálních a anomálních videí. Jedná se o stejnou sadu, jako je popsána v kap. 5.1.1. Za zásadní se dají považovat videa anom-4 a anom-5.

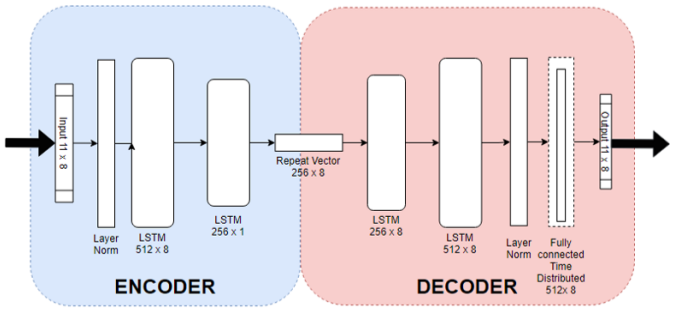
Pro připomenutí: v prvním případě je video natočeno z mírně jiného úhlu, zastoupeného pouze minoritně v datové sadě. V případě druhém jde navíc o řidiče, který se v datové sadě nevyskytuje. Pro nové typy úhlů je příprava datové sady a dávky stejná, jako v kapitole 5.1.1., protože tyto úhly jsou vytvořené stejným způsobem. Není třeba je znovu extrahovat, protože, během původní konstrukce byla vytvořena daleko větší množinu příznaků, než bylo v základním navrhovaném řešení použito, stačí tedy zvolit jinou podmnožinu úhlů a tu nakonec transformovat do vstupního formátu neuronové sítě - $8 \times 8 \times 11$.
Pro sadu příznaků tvořenou klíčovými body je třeba pouze provést jejich extrakci z videozáznamů pomocí systému Open-Pose v normalizovaném formátu. To znamená, že systém transformuje jejich souřadnice do rozsahu -1 a 1. To je provedeno aby se zmenšil vliv, rozdílů ve formátu videa a pozice na výslednou sekvenci. Výstupní data je třeba také transformovat do formátu 8 x 8 x 25, obdobně viz výše. Body x a y jsou v jednotlivých dávkách příznaků vedle sebe.

7.1.2 Hledání optimálních hodnot hyperparametrů sítě pro řešení využívající alternativních klíčových bodů

Architektura sítě vychází pro oba druhy příznaků ze základního řešení. S ohledem na jiné zjištěné závislosti a formát však muselo být řešení pozměněno. Jde především o použití aktivační funkce *relu* a beze-stavové LSTM (statefull parametr byl vypnut). To dovoluje lépe optimalizovat síť. Výsledky nebyly příliš dobré, pokud byla použita normalizační vrstva za každou LSTM, proto byla zaměněna za normalizaci pouze na začátku a konci sítě podél osy -1. Normalizace byla tedy prováděna pro poslední dimenzi. To pomohlo výrazně zrychlit a stabilizovat trénink.
Pro velikost vstupní dávky LSTM a mezery mezi snímky byly ponechány stejné hodnoty jako je tomu u základního navrhovaného řešení. Jednou ze změn je zvětšení batch-size, jak pro příznaky využívající klíčových bodů, tak pro nové typy úhlů. Hodnota 8 použita v základním navrhovaném řešení nedovolovala dosáhnout optima u tréninkového procesu. Optimální hodnota byla v tomto případě 256 pro příznaky tvořené body a 64 pro novou selekci úhlů. Snižování zhoršovalo výsledky tréninku - a ani následné zvyšování nepomohlo k žádnému zlepšení, pro obě konfigurace. Příčinou nutných změn je s největší pravděpodobností jiný druh závislosti, než jakou příznaky reprezentují, a proto je obtížné najít optimální hodnotu.
Změna počtu vrstev nebo výrazná změna počtu jednotek v jednotlivých vrstvách nijak nezlepšily validační výsledky během úvodních testů. Počet vrstev byl proto ponechán na hodnotě 2 pro každou část a iterovány byly počty jejich jednotek stejně jako tomu bylo v základním navrhovaném řešení. Ve finále byla použita konfigurace sítě, viz ob. 7.1.

7.1.3 Způsob vyhodnocení výsledků

Postup je stejný jako v kapitole 5.1.3. Každý běh pro každou architekturu je spuštěn třikrát. Nejdříve je provedeno hledání optimální architektury pomocí validačních sad anom-1, anom-2 a anom-3. Po

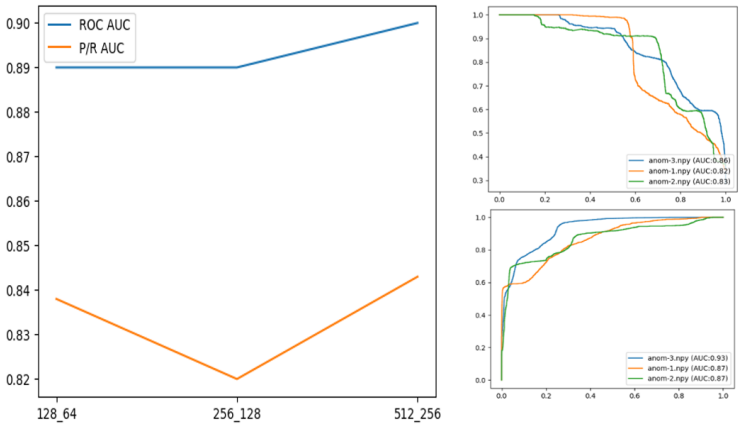


Obrázek 7.1: Architektura řešení využívající alternativní obrazové příznaky

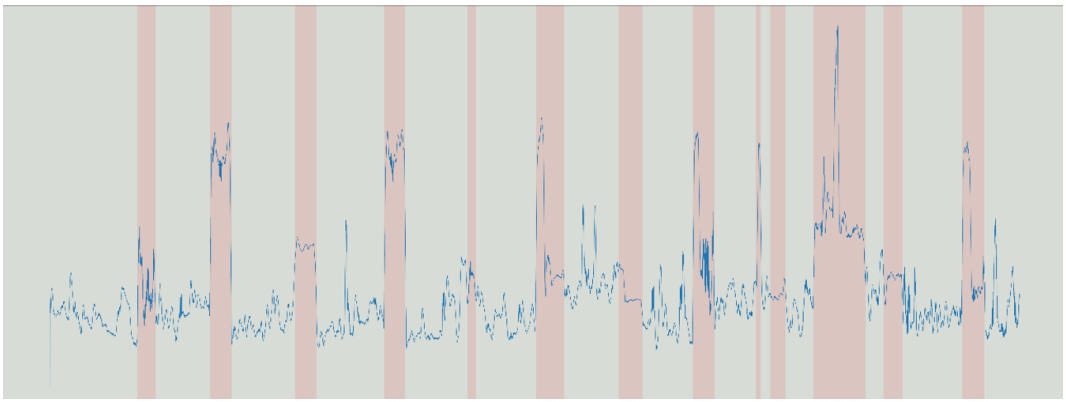
uložení P/R a ROC charakteristik, dochází k výpočtu průměrné hodnoty pro každou velikost sítě. Z nich je následně vybrán nejlepší výsledek, a ten pak zhodnocen pomocí řečných datových sad. Pomocí sady anom-4 dochází k porovnání možnosti detekce v jiném prostředí, a se sadou anom-5 jsou vyhodnoceny vlastnosti týkajícími se neznámého řidiče. Nyní následuje prezentace výsledků experimentů.

7.2 Hodnocení výsledků příznaků tvořených alternativními úhly

Pro alternativní úhly jsou výsledky detekce srovnatelné se základním řešením. Při srovnání je patrné, že nejlepších výsledků dosahuje opět architektura o velikost 512/256. Je patrná velice čistá a stabilní detekce anomálií, bohužel ani změna úhlů nijak nepřispěla k detekci těžko zachytitelných anomálií. V případě například anomálie #3, tomu bylo právě naopak. Důvodem je patrně jejich nízká rekonstrukční chyba, způsobena strnutím řidiče. Je pozorovatelné, že anomálie tohoto druhu jsou autoenkodérem rekonstruovány efektivněji.

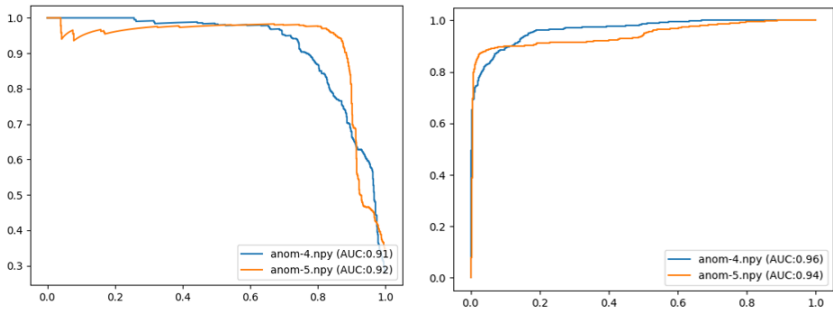


Obrázek 7.2: Výsledky P/R a ROC optimální architektury pro alternativní sadu úhlů



Obrázek 7.3: Hodnota rekonstrukční chyby anom-3 pro řešení využívající alternativní sadu úhlů

Toto řešení dosahuje téměř perfektních výsledků, a to jak pro validační sadu s viděným, tak pro validační sadu s neviděným řidičem. Charakteristiky naznačují, že došlo k detekci naprosté většiny anomálií. ROC křivka i P/R křivka dosahují více než 90 %, výsledky jsou srovnatelné a v mnoha bodech překonávající základní řešení.

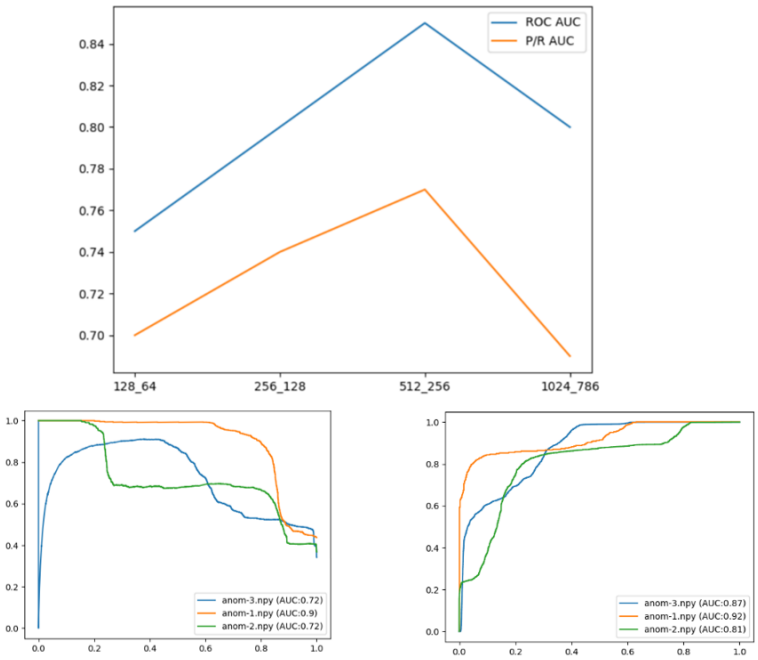


Obrázek 7.4: Výsledky P/R a ROC anom-4 a anom-5 pro řešení využívající alternativní sadu úhlů

7.3 Hodnocení výsledků příznaků tvořených klíčovými body

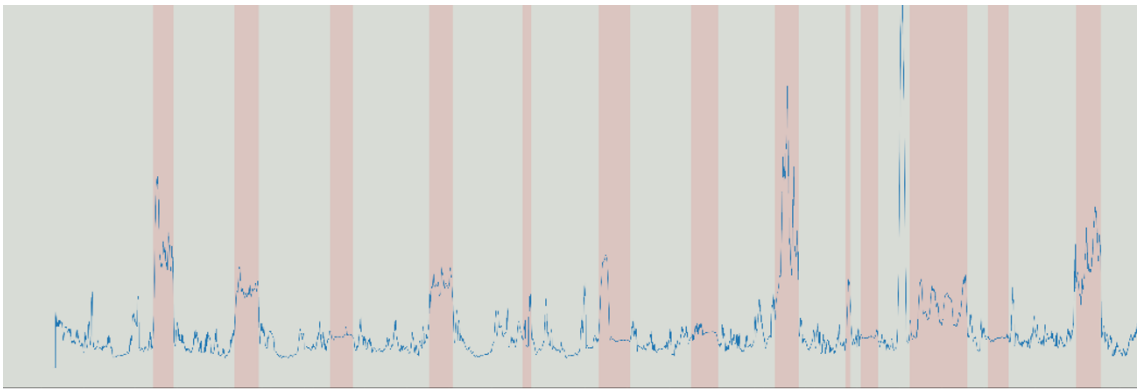
Nejlepších výsledků dosáhla implementace o rozměrech 512/256. Průměrná přesnost se pohybovala kolem 78%. Ani po poměrně dlouhém iterativním prohledávání hyper-parametrů a odstranění normalizace nebo po provedení změn v architektuře sítě se tyto výsledky nepovedlo příliš vylepšit. Jak ROC, tak P/R křivky poukazují na problémy způsobené tímto výběrem obrazových příznaků. Experimentální výsledky, potvrzují předpoklad, že úhly, jsou vhodnější reprezentací pro tento problém než klíčové body.

Příčina je opět patrná z charakteristiky ROC. Pro velice čistou validační sadu anom-1 dochází k velice dobré detekci většiny anomálií. Mírné stoupání indikuje pouze nedokonalé zachycení některých z nich. Zlom viditelný kolem hodnoty 0.1 pro FPR ukazuje, že se nepovedlo zachytit úplně všechny



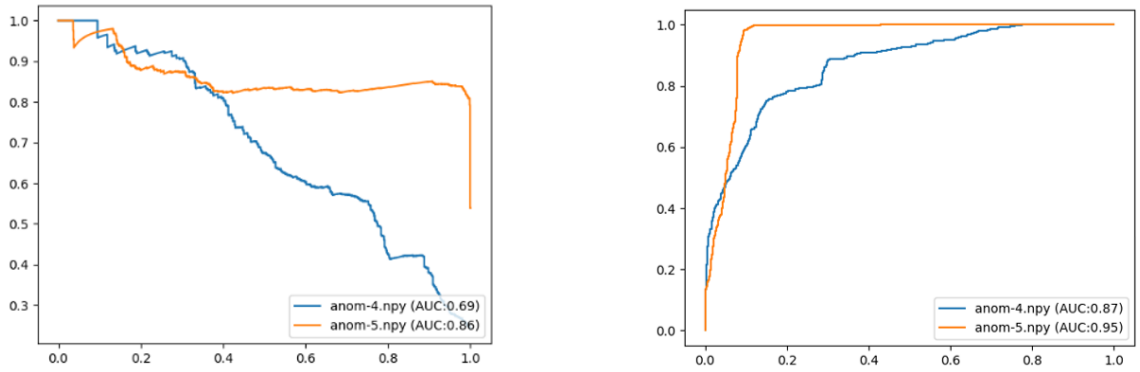
Obrázek 7.5: Výsledky P/R a ROC pro optimální architekturu využívající klíčových bodů

anomálie. Naproti tomu u validačních sad anom-2 nebo především u validační sady anom-3 dochází k několika zlomům. Detekce zhruba 60% anomálií pro sadu anom-3 je velice problematická. Pokud je cílem dosáhnout přesnosti alespoň 60%, pak je zřejmé, že bude možné detekovat pouze zhruba 50% anomálií. Na obrázku 7.6 je znázorněna chybová funkce validační sady anom-3. Jak je vidět, problematické anomálie popsané v kapitole 5.2 zůstávají problematickými. Navíc se zhoršuje detekce pro některé další, jako například #11. Naproti tomu chybová charakteristika vykazuje naprosté minimum šumu, čímž jednoznačně předčí základní navrhované řešení.



Obrázek 7.6: Chybová funkce validační sady anom-3 pro řešení využívající klíčových bodů

Nečekaně dobrých výsledků bylo dosaženo u validační sady č. 5. Naměřené hodnoty ukazují jistou schopnost generalizace v rámci neviděných řidičů. Současně byla odhalena jistá nevýhoda, totiž snížená schopnost rozpoznat řidiče podle tělesných proporcí. Figurantka z videa anom-4 je poměrně netypická, protože její výška a způsob držení těla jsou pro datovou sadu extrémní. V porovnání s výsledky základního řešení je viditelné, že některé anomálie se nepovedlo příliš dobře rekonstruovat. To ukazuje na nevýhodnost tohoto přístupu pro malé datové sady, s malým vzorkem řidičů. Je výhodné využít jistého druhu augmentace dat. Tím je myšleno různým způsobem deformovat proporcionálně výsledné pozice bodů, což může zlepšit konečné vlastnosti.



Obrázek 7.7: Výsledky P/R a ROC anom-4, anom-5 pro řešení využívající klíčových bodů

7.4 Diskuse

Série provedených experimentů porovnávala vlastnosti řešení v případě využití jiné formy obrazových příznaků. Přestože muselo být přistoupeno k mírným úpravám sítě, je zřejmé, že navržená architektura se hodí pro řešení tohoto problému. Je otázkou diskuse, jestli by bylo možné najít lepší množinu příznaků. To by se samozřejmě dalo vyřešit optimalizační cestou, nicméně experiment takového rozsahu je zcela nad rámec této práce.

Experiment potvrzuje úvahu o vyšší vhodnosti úhlů ve srovnání s klíčovými body. Tento přístup má značný potenciál v mnoha oblastech závislých na rekonstrukci pózy. Přesnost systémů jako je Open-Pose nabízí tuto možnost jako alternativu k již používaným metodám v analýze obrazu.

Současně provedený experiment nepotvrdil hypotézu z kapitoly 5.3. - navržená variace úhlů nepřináší znatelně lepší výsledky pro těžko detekovatelné anomálie. Detekční výsledky těchto anomálií jsou srovnatelné se sadou navrženou v kapitole 4.1.2. Vylepšením by byla kombinace s úhly rozsahu 0°-180°, ale pouze pro vybrané z nich (hlavy a ramen). Jak bylo naznačeno v kapitole 4.1.2, v případě použití pouze úhlů v rozsahu 0°-180° systém dosahoval znatelně horších výsledků. Příčinou je nejspíše dvojnásobná velikost rozsahu pro jednotlivé úhly, a tedy složitější reprezentace závislostí pomocí neuronové sítě.

Detekce u obou variant je nepatrně méně efektivní, ale jejích chybová charakteristika se zdá být daleko čistší, s menším množstvím šumu. Je možné, že příčinou nižší citlivosti je paradoxně řečený nižší šum a tím pádem také čistší detekce jednotlivých anomálií. Anomálie lze na grafu chybové funkce pozorovat jako prudké vzestupy hodnoty. Pokud autoenkodér dokáže rekonstruovat hodnotu bez potíží, hodnota chyby bude malá, ale také bude nejspíše malá hodnota šumu. Pokud tedy změnami v tréninkovém procesu nebo architektuře dojde ke zvýšení šumu, jeho zvýšení bude více patrné na místech, kde měl autoenkodér problémy rekonstruovat vstup, což budou právě zmíněné anomálie. Je tedy možné, že stabilní hodnota chybové funkce, paradoxně ztěžuje proces detekce chyby, na druhou stranu je možné takto stabilní hodnotu upravit s využitím vhodné funkce.
Funkce by prováděla úpravu hodnoty chyby pomocí posuvného okna. Okno by na základě změny chybové hodnoty (první derivace), tam kde již delší dobu nedošlo k její minimální průměrné změně, tuto hodnotu násobilo. To by sice zvýšilo hodnoty chyby globálně, ale primárně v místech, kde řidič strnul, tedy i v případě kolapsu nebo mikrospánku.
Dalším krokem je experimentace s architekturou sítě, nad rámec úpravy hyper-parametrů a počtu vrstev. LSTM autoenkodér se snaží komprimovat vstupy a efektivně z nich rekonstruovat výstupy. Tyto vlastnosti jdou proti sobě. Větší neuronová síť bude schopna rekonstruovat složitější sérii vstupů, ale nebude nucena vytvářet efektivní kompresi v úzkém hrdle. Menší bude komprimovat efektivně, ale bude mít potíže s rekonstrukcí. Pravděpodobně existuje optimum, vyvážený bod, který se však velice těžko hledá.
Další experimenty se tedy věnují úpravám architektury sítě, které by mohly vést k lepším výsledkům. Tyto experimenty budou opět vycházet ze základního navrhovaného řešení.

Kapitola 8

Vylepšení architektury sítě základního řešení

Základní architektura je tvořená téměř čistě LSTM vrstvami, z toho ale plynou jisté nevýhody. Úzké hrdlo, které komprimuje vstupní data, je realizováno poslední LSTM vrstvou, což činí enkodér a dekodér strukturálně odlišnými a fakticky zmenšuje počet vrstev enkodéru, které pracují se všemi skrytými stavy. Tato kapitola se zabývá návrhem alternativních řešení a vylepšením tohoto návrhu. Jako první je uveden popis alternativní implementaci čistě pomocí fully-connected vrstev. Následuje implementace rozšíření základního řešení, kdy jsou rovněž přidány další fully-connected vrstvy a následně podobným způsobem rozšířená implementace s využitím attention vrstvy nahrazující úzké hrdlo.

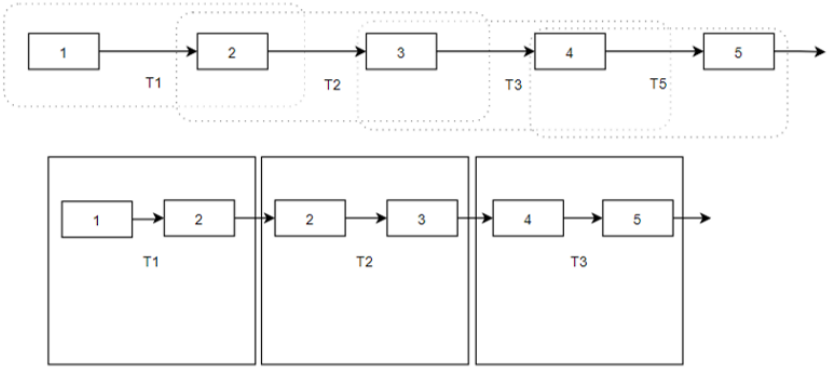
Tato kapitola je členěna následujícím způsobem:

- **Kapitola 8.1** Implementace fully-connected autoenkodéru.
- **Kapitola 8.2** Rozšíření základního řešení pomocí fully-connected vrstev.
- **Kapitola 8.3** Rozšíření základního řešení pomocí attention vrstev.

8.1 Implementace autoenkodéru pomocí fully-connected vrstev

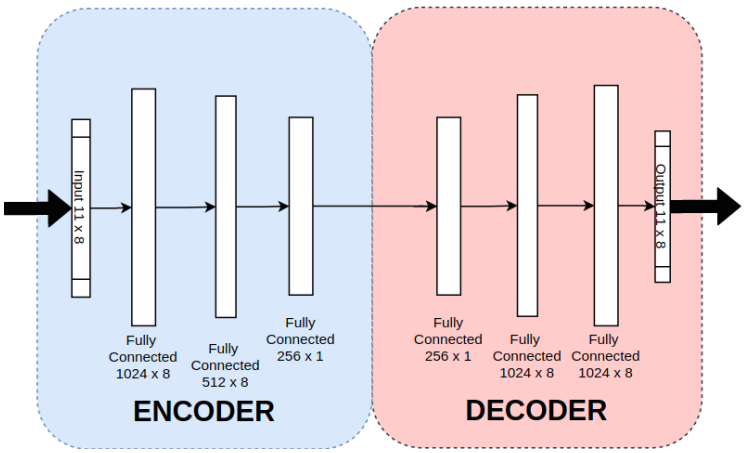
Teoreticky je možné implementovat autoenkodér pomocí fully-connected vrstev. Tato implementace sice zanedbává časové závislosti nad úroveň velikosti dávky, nicméně by to nemuselo v tomto případě vadit. Jednou z alternativ je transformace vstupních dat pomocí pohyblivého okna pevné délky. Princip je podobný jako u inference v kapitole 4.4.1. Skrze vstupní vektor příznaků vybraných z jednotlivých po sobě jdoucích, chronologicky seřazených snímků je posouváno okno o velikosti vstupní dávky d_i. V každém čase t je posunuto o vybraný počet snímků. V čase t_n okno kopíruje do výstupního vektoru data od indexu t_n po index $t_n + d_{i-1}$. Následně je okno posunuto o daný

počet snímků. Takto postupně dojde ke kopírování celého vstupního vektoru. Během tréninku tak v každém čase neuronová síť přijme vstupní dávku posunutou o daný počet snímků, což vytváří závislost mezi jednotlivými vstupními dávkami.



Obrázek 8.1: Proces pohyblivého okna

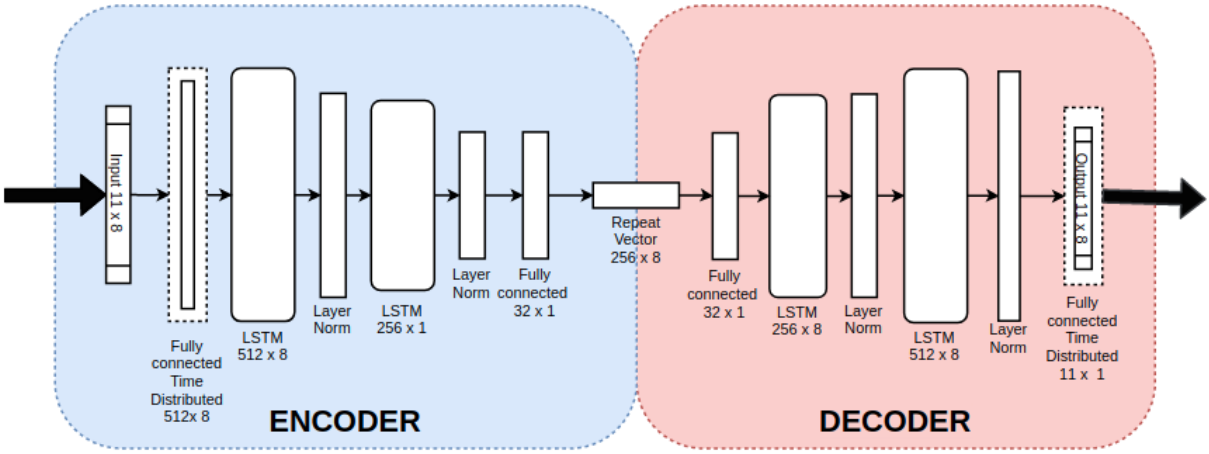
Nejprve bude proveden pokus srovnat klasickou standardní fully-conected síť s datovou sadou tvořenou klasickým vstupním vektorem jako u LSTM řešení a také popsanou variantou pomocí posuvného okna. Experiment bude využívat tři vrstvy, protože optimální hodnota pro základní řešení nepodávala během přípravy uspokojivé výsledky. Hledání optimálního počtu jednotek pro jednotlivé vrstvy je prováděno iterativně jako u základního řešení. Nejefektivnější architektura je zobrazena níže. Pro obě varianty je nutné změnit velikost vstupní dávky na 16, velikost mezery mezi snímky zůstala na mod 3. Batch-size bylo třeba zvětšit na 512, jinak nedocházelo k dobrým výsledkům tréninkového procesu. Trénink sítě byl nejefektivnější do dvou epoch, vyšší počet pouze zhoršoval validační výsledky. Stabilita tréninku mezi jednotlivými nastaveními náhodného generátoru byla o něco nižší.



Obrázek 8.2: Architektura alternativního řešení využívající pouze fully-conected vrstvy

8.2 Modifikace autoenkodéru pomocí fully-conected vrstev

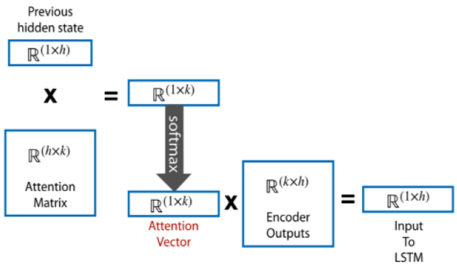
Tato část experimentu se zaměřuje na vylepšení vlastností úzkého hrdla autoenkodéru přidáním dalších fully-conected vrstev. To tvoří poslední skrytý stav poslední LSTM, který je opakovaně pomocí tzv. repeat-vektoru, zasílán tak, aby měl jeho výsledek dimenzi shodnou se vstupní dimenzí dekodéru. Poslední LSTM vrstva enkodéru tedy nevrací celou sekvenci zakódovaných výstupů, což zajišťuje kompresi informace.
Experimentálně jsou přidány další fully-conected vrstvy na obě strany úzkého hrdla. Jelikož poslední vrstva LSTM enkodéru vrací pouze poslední skrytý stav, není nutné využívat time-distributed vrstvy pro propagaci vah skrze čas. Velikost nové vrstvy bude výrazně menši, než předchozí LSTM aby byla síť během tréninku nucena vytvořit, co možná nejefektivnější representaci vstupní sekvence.
Další úpravou je přidání fully-connected vrstvy na začátek, což vyváží vstupy před jejich posláním do první LSTM vrstvy. Velikost nové vstupní fully-connected vrstvy bude stejná jako je tomu u poslední vrstvy dekodéru. Tato vrstva je také obalena time-distributed vrstvou, aby byla zachována dimenze vstupu. Normalizační vrstvy byly umístěny shodně se základním řešením. Velikost batch-size a formát vstupní dávky jsou shodné jako u řešení popsaného v kap. 5.1.2., počet epoch je nastaven na 30. Optimální hodnoty počtu jednotek jsou zobrazeny níže.



Obrázek 8.3: Architektura alternativního řešení využívající upraveného autoenkodéru

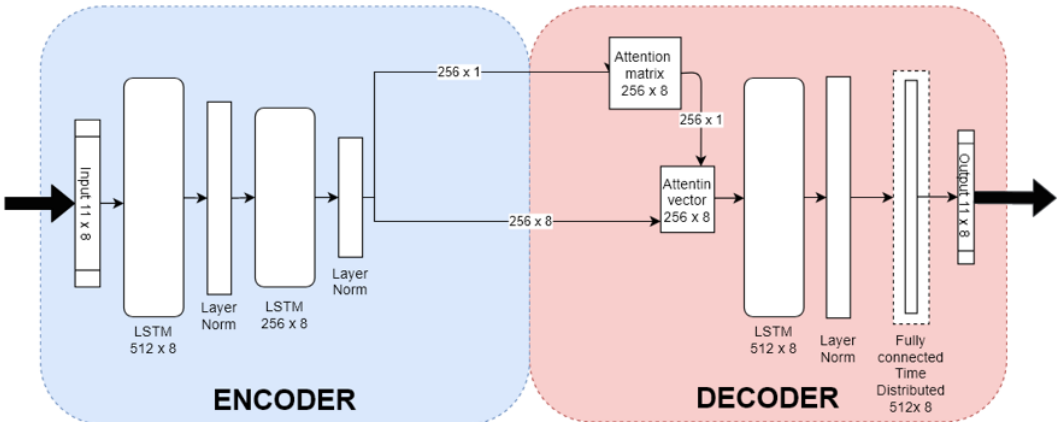
8.3 Náhrada úzkého hrdla attention vrstvou

Jednou z možností, jak modifikovat autoenkodér je přidání attention vrstvy. Attention mechanismus byl již popsán v kapitole 3.2.1. Implementace, ze které je v práci vycházeno, byla popsána ve studii A3D[10]. Principiálně se jedná o tzv. Luong-Attention[11]. Attention dovoluje autoenkodéru se soustředit pouze na některé hodnoty ve vstupní sekvenci a snaží se o jejich vyvážení. Mechanismus funguje na principu prostého násobení a je popsaný na obrázku níže. Je zřejmé, že nedochází ke standardní kompresi sekvence, protože nedochází k žádnému zúžení v podobě redukce dimenze výstupu poslední LSTM vrstvy.



Obrázek 8.4: Attention mechanismus ve A3D [10]

Shodně se studií struktura vrstev nekopíruje architekturu autoenkodéru, ale nahrazuje první LSTM vrstvu dekodéru za již řečenou attention matici. Bylo experimentováno i s variantou, kdy je attention vrstva přidána pouze jako alternativa úzkého hrdla a počty vrstev a jednotek pro obě části autoenkodéru zůstávaly stejné. Dosažený výsledek byl mírně horší.
Experimentální implementace je založena na základním řešení. Většina hyperparametrů jako je velikost vstupní dávky, mezera mezi snímky nebo batch-size jsou pro tento experiment stejné. Normalizační vrstvy jsou umístěny shodně se základním řešením. Počet tréninkových epoch byl pro tento experiment 30. Nejefektivnější architektura je vyobrazena na obrázku níže.



Obrázek 8.5: Alternativní architektura autoenkodéru využívající attention mechanismu.

Kapitola 9

Experimentální realizace vylepšení architektury sítě základního navrhovaného řešení

Tato kapitola se zabývá experimentální realizací vylepšení architektury základního řešení popsaného v kapitole 8. Experimenty jsou zaměřeny krom vylepšení architektury řešení a ověření jeho vhodnosti také na srovnání variant a jejich vlastností. Nejprve jsou popsány možnosti plynoucí z implementace využívající pouze fully-connected vrstev. Zde je následně experimentováno také se vstupem v podobě pohyblivého okna. Následuje rozšíření základního řešení o vhodné prvky, v tomto případě fully-connected vrstvy, s cílem vylepšení vlastností komprese vstupu. Nakonec je provedeno srovnání celé implementace s řešením založeným na mechanismu attention, kterým je nahrazeno úzké hrdlo autoenkodéru.

Toto řešení je strukturováno podobně jako zbylé experimenty:

- **Kapitola 9.1** se zabývá metodikou jednotlivých experimentů.
- **Kapitola 9.2** popisuje vyhodnocení výsledků.
- **Kapitola 9.3** je určená diskusi a vyhodnocení závěrečných výsledků.

9.1 Metodika realizace experimentálního řešení

Datová sada se skládá ze stejných videozáznamů, definovaných v kapitole 5.1.1, pro tréninkovou i validační sadu. To stejné platí pro sérii vstupních příznaků a mezeru mezi snímky. Hledání architektury je provedeno u každého z navrhovaných řešení zvlášť. V kapitolách níže jsou popsány jednotlivé návrhy.

9.1.1 Hledání optimální architektury řešení používající pouze fully-connected vrstvy

Na tomto místě byl experiment proveden se dvěma variantami vstupů. U první varianty používající standardní vstupní vektor, přetransformovaný dle pravidel definovaných v 5.1.1, nebylo třeba provádět žádné větší změny v architektuře sítě. Tentokrát byly zvoleny tři vstupní vrstvy namísto dvou. Rovněž byl použit větší počet jednotek jednotlivých vrstev od 1024 níže, protože fully-connected vrstvy nedokážou reprezentovat časové závislosti a potřebují proto větší počty jednotek, než LSTM, aby byly schopny vytvořit dostatečnou reprezentaci vstupu. Další zvětšování velikosti sítě k lepším výsledkům nevedlo.
Pro experiment bylo rovněž nezbytné zvětšit velikost vstupní dávky na 16, protože velikost použitá v základním řešení nedosahovala příliš dobrých výsledků. V případě využití pohyblivého okna bylo nutné změnit vedle dávky také princip její přípravy. Pohyblivé okno vede k nárůstu délky vstupního vektoru. Při výzkumu se opakovaně vyskytla situace, kdy nebylo možné pro Tensorflow alokovat dostatek paměti. Byla použita pouze první série snímků mod $3 = 0$, jak je popsáno v kapitole 4.2. Paradoxně docházelo k přeučení sítě i během jedné epizody, což by se sice dalo řešit další modifikací tréninkového procesu, nicméně to nebylo nutné, protože i tak výsledky dokazují, že tato implementace v mnohém předčí LSTM. Velikost batch-size byla 512 a počet epoch nakonec 2. Porovnání architektur a výsledků bylo provedeno iterativně, stejným způsobem jako v kapitolách 5.1.2 nebo 7.1.2.

9.1.2 Hledání optimální architektury vylepšení autoenkodéru fully-connected vrstvami

U tohoto řešení byly zvoleny stejné parametry jako popsané v základním navrhovaném řešení v kapitole 5.1.2. Protože je cílem zlepšit vlastnosti komprese informací, střední fully-connected vrstva byla nastavena na hodnotu od 32 do 128 jednotek. Změny v počtu jednotek byly nejpatrnější právě v případě této vrstvy. Zvětšení nebo zmenšení počtu jednotek mimo tento rozsah k žádným zásadním zlepšením nevedlo. Ostatní hodnoty hyperparametrů zůstaly ponechány stejné jako v kap. 5.1.2. Porovnání výsledků bylo prováděno stejně jako u předešlých pokusů.

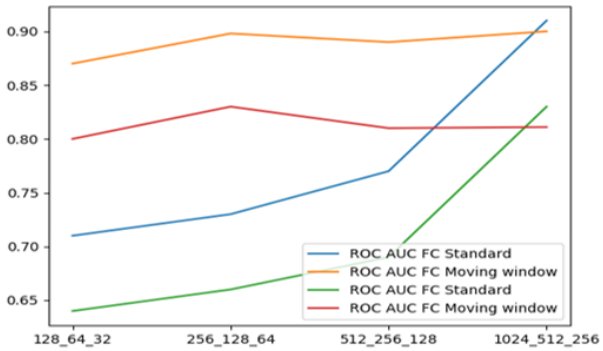
9.1.3 Hledání optimální architektury řešení používající attention vrstvy

Popis hledání optimální architektury je v tomto případě velice podobný předchozímu příkladu. Mechanismus popsaný v 3.2.1 a 8.2 přinášel nejlepší výsledky pro většinu pokusů. Během experimentace bylo iterováno především s počtem vrstev, jak je popsáno v kapitole 8.3. Výsledky dvou a tři vrstev pro dekodér jsou srovnatelné.

9.2 Vyhodnocení výsledků řešení využívající jen fully-connected vrstev

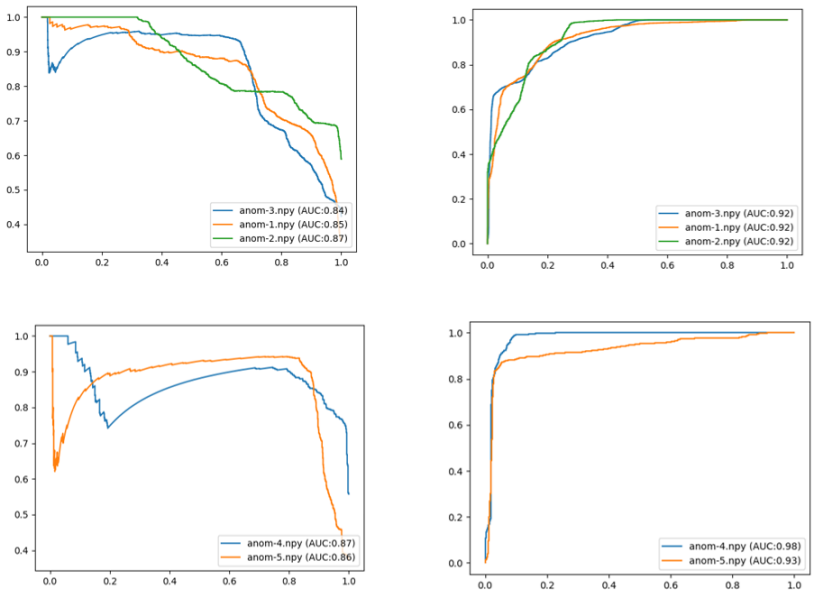
Implementace pomocí pouze fully-connected přinesla překvapivé výsledky. Po zvýšení batch-size, počtu vrstev a jejich jednotek bylo dosaženo výsledků, které jsou srovnatelné se zbytkem experi-

mentů. Největší výhodou je opravdu vysoká rychlost tréninku. Jak bylo uvedeno v kap. 9.1.1, stačily pouze dvě epochy. Pro použití pohyblivého okna bylo nutné pozměnit proces dávky. Proto nelze vyloučit, že při jeho další úpravě a přidání dalších dávek by byly výsledky ještě o něco lepší.

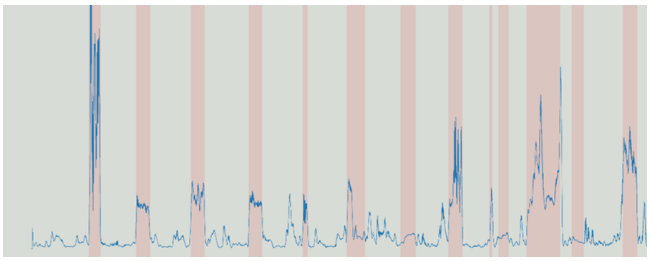


Obrázek 9.1: Výsledky alternativní realizace využívající fully-connected architektury

AUC charakteristiky pro ROC i P/R ukazují detekci většiny anomálií. Je překvapivé, že byly detekovány i anomálie, se kterými měly jiné varianty experimentu potíže. Především úroveň rekonstrukční chyby anomálie #3 je srovnatelná s anomáliemi #2 a #4, což se u jiných experimentů nepovedlo. V případě anomálií #7, #10 a #12 k téměř žádné rekonstrukční chybě nedochází.



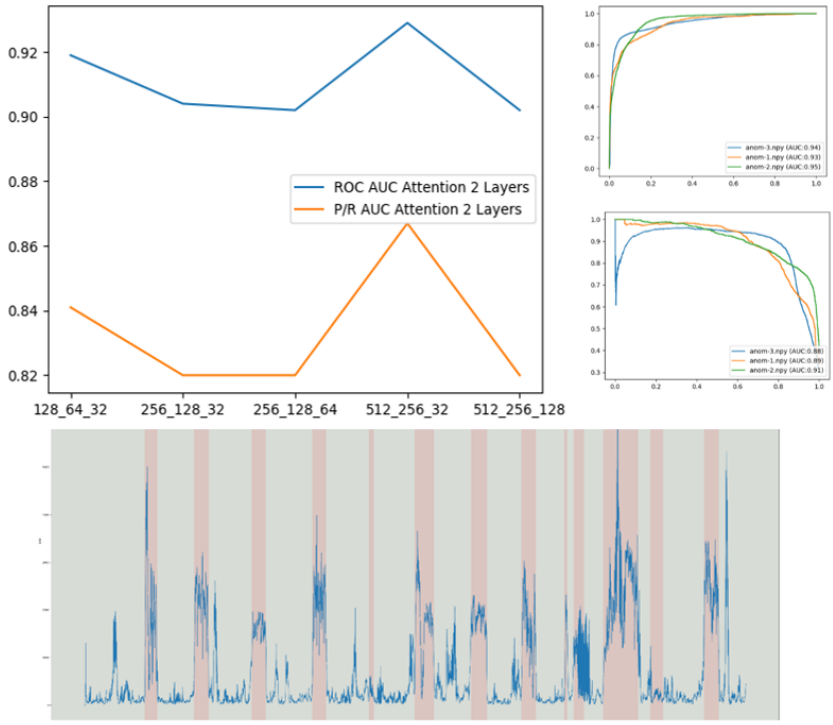
Obrázek 9.2: P/R a ROC křivky realizace využívající fully-connected architektury



Obrázek 9.3: Chyba pro validační sadu anom-3 realizace využívající fully-connected architektury

9.3 Vyhodnocení výsledků vylepšení LSTM autoenkodéru

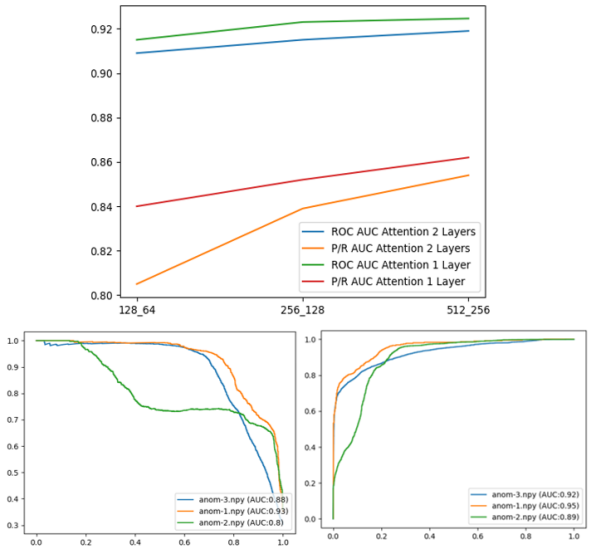
Výsledky LSTM autoenkodéru mírně předčí základní řešení. Ve srovnání s předchozím experimentem je vyšší nejen průměrná přesnost, ale také počet epoch a celková doba tréninku. Zlom v 0.8 u validační sady anom-3 ale ukazuje, že nebyly korektně detekovány všechny anomálie (konkrétně se jedná o #4 a #11). Nejlepších výsledků dosahovala architektura o velikost 256/128/32. Obdobných výsledků bylo dosaženo již u architektury 128/64/32. Nejdůležitějším parametrem je evidentně velikost úzkého hrdla, která je efektivní od 32 jednotek. Poměrně velký šum u anomálie #10 je způsoben poněkud neobvyklou normalizací podél osy 1. Na druhou stranu se s jejím využitím zvýšila hodnota ROC AUC zhruba o 5 %.



Obrázek 9.4: Výsledky alternativní realizace využívající změněné autoenkóder architektury

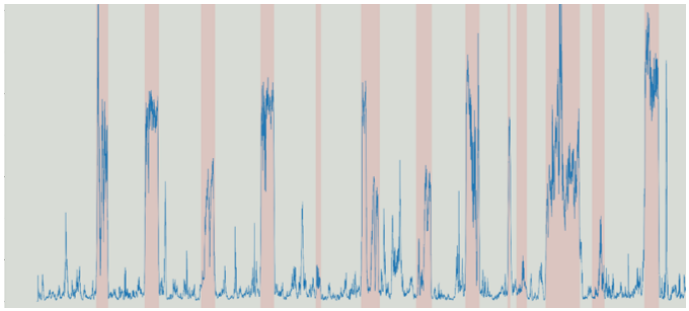
9.4 Vyhodnocení výsledků řešení využívajícího attention mechanismu

Autoenkodér využívající attention mechanismus dosáhl zatím nejstabilnějších výsledků z testovaných architektur sítě. Je zřejmé, že byť jde pouze o jistou formu úpravy úzkého hrdla implementace, schopnosti detekce jsou, ale výrazně zvýšeny. Během experimentace byly srovnány varianty se dvěma a jednou skrytou vrstvou pro dekodér. Výsledky jsou srovnatelné a rozdíly zanedbatelné. Mírně lepších výsledků dosahuje varianta s jednou vrstvou, ale jde opravdu o desetiny procenta.



Obrázek 9.5: Porovnání P/R a ROC realizace využívající attention mechanismu

Z grafu chyby je viditelné zlepšení schopnosti detekce, velké zmenšení šumu a vysoká přesnost ohraničení anomálií. Nejlépe si vedla varianta s 512/256 jednotkami. Problémy způsobenými normalizací tato architektura netrpí. Přesto se nepovedlo zlepšit úroveň chyby pro chyby #5, #12 a paradoxně #10, jejichž detekce je nyní slabší.



Obrázek 9.6: Hodnoty chybové funkce pro anom-3 realizace využívající attention mechanismu

9.5 Diskuse

V prvním experimentu (viz. kap. 9.2) byl autoenkodér realizován pouze pomocí fully-connected vrstev. Je vidět, že tato implementace je při vhodné přípravě datové sady velice efektivní alternativou jakékoli LSTM implementace. Obrovskou výhodou je rychlost tréninku a možnosti paralelizace procesu, kterou LSTM neposkytují. Nevýhodou je relativní nestabilita, která by se ale dala vylepšit přidáním vhodné normalizace a případného klipování gradientu.
V experimentu 9.3 je vidět, že upravený autoenkodér tvořený LSTM a fully-connected vrstvami má vůči původní realizaci základního řešení značné množství výhod a je tedy, preferovaným řešením v případě snahy o použití LSTM autoenkodéru. Největší výhodou je stabilita tréningu a potencionálně malé rozměry sítě. Je pravděpodobné, že celkové výsledky této implementace jsou silně ovlivněny vybranými hodnotami hyperparametrů a bylo by možné jeho efektivitu dále zvýšit. U anomálie #4 je viditelně zvýšená úroveň šumu. Tento problém lze vyřešit náhradou za sofistikovanější implementaci normalizace dávky, která nebyla k dispozici v použité verzi knihovny Tensorflow a její dodatečná implementace je mimo rozsah této práce. Tento šum vzhledem k úrovni detekce anomálie #4 v kapitole 9.4 potvrzuje závěr diskutovaný v kapitole 7.4 - anomálie mají vzhledem k principu autoenkóderu velice často vyšší úroveň šumu, než jak tomu je v případě normálních dat.
V experimentu 9.4 je viditelné, že attention vrstva je použitelná jako dobrá alternativa úzkého hrdla. Je schopna vytvořit efektivní reprezentaci, i když nedochází k velkému zúžení, a tedy zásadní kompresi informací. Dalším logickým krokem je cesta směrem k transformer architekturám. Ty využívají bloky self-attention, které by mohly tento problém řešit velice efektivně. Pokud by byla tato implementace navíc tvořena pouze fully-connected vrstvami, jak je tomu u například GPT 3[34], poskytovalo by to několik výhod. Nejzásadnější z nich je paralelizace tréninku. Také by bylo možné teoreticky řešit detekci anomálií jako dvoutřídní klasifikační problém. Pokud si bude transformer jist svou předpovědí, budou data normální, v opačném případě anomální. U výstupního formát by pak naproti autoenkodéru nebylo potřeba vyhodnocovat úroveň chyby, což pramení z jiného principu fungování.
Nevýhodou by byla velká náročnost tréninkového procesu. Zde by byla zapotřebí opravdu velká datová sada. Výběr těch nejlepších a nejefektivnějších příznaků může být obtížný. Transformerové architektury jsou navíc velmi náročné z hlediska vývoje i ceny, proto experiment tohoto typu není v rozsahu této práce.

Kapitola 10

Závěr

Hlavním přínosem této práce je především kombinace sběru a převodu obrazových dat na klíčové body, jejich reprezentace a následné vyhodnocení anomálnosti pomocí neuronových sítí. Byly ověřeny teoretické předpoklady o fungování, problémech a vlastnostech takovéhoto systému.
Je zřejmé, že využití neuronových sítí v kombinaci s variantami příznaků tvořenými pomocí klíčových bodů je velice efektivní cestou, jak detekovat kritické - potenciálně fatální - situace, k nimž může dojít při řízení vozu. Při této práci vlivem koronavirové epidemie nebyla k dispozic příliš velká datová sada a sběr dodatečných dat byl limitován na lokální simulační prostředí. I přes tento nedostatek, se povedlo experimentálně realizovat základní navrhované řešení. Následně byla ověřena efektivita úhlů tvořených klíčovými body a zároveň replikovány výsledky základní navrhované realizace s využitím pozměněné reprezentace obrazových příznaků. Tuto architekturu se povedlo vylepšit. Byla transformována pomocí fully-connected jednotek do symetrického, pro autoenkodéry typického, tvaru sítě. Také bylo experimentálně ověřeno, že je možné realizovat tuto architekturu čistě pomocí fully-connected jednotek. I když má tato varianta několik nevýhod, je především díky rychlosti, efektivní variantou realizace. Nakonec byla realizována implementace pomocí attention vrstvy, která se ukázala jako vcelku velice efektivní varianta standardní autoenkodér architektury.

U provádění experimentů, byly dále zjištěny dodatečná vylepšení navrhované realizace. Z kapacitních a časových důvodů nebyla možná jejich přímá implementace. Především by bylo vhodné experimentovat se tvarem a typem chybové funkce. Toho může být dosaženo např. modifikací popsanou v kap. 7.4. - nebo náhradou za zcela odlišnou funkci. Dalším vylepšením by pak byla experimentace podobná té z kapitoly 9.1.3 s využitím několika attention hlav. Teoreticky také s využitím pouze fully-connected vrstev. Tato architektura by byla podobná, ne však shodná s transformer architekturou.
Další vývoj v této oblasti bude nejspíše následovat trend v podobě využití zmíněné transformer architektury. Fully-connected vrstvy napomáhají rychlejšímu tréningu. Attention matice jsou schopny efektivně komprimovat informaci a využití několika hlav. Jejich využití proto může vést k velice dob-

rým výsledkům pro extrémně velké datové sady. Dalším nadějným krokem je pak experimentace v oblasti kompozice transformeru z menších, specializovaných částí. Velkou výhodou je v tomto ohledu velice dobrá vizualizace vnitřního rozhodování a fungování jednotlivých attention hlav. Výsledný systém by pak byl modularizovatelný a vylepšitelný přímo v provozu, což z něj dělá velmi efektivní a bezpečnou metodu.

Pro samotnou realizaci tohoto řešení je potřeba zajistit bezpečný a anonymní sběr dat přímo za provozu pro jednotlivé řidiče. Toho lze dosáhnout s využitím systémů, jako je např Open-Pose. Klíčové body naprosto přesně popisují, co se v automobilu děje, navíc nijak nenarušují anonymitu jednotlivých řidičů. Právě tyto vlastnosti by mohly znamenat velkou výhodu, nad variantami vyžadujícími komplexnější obrazové příznaky.

Literatura

1. ZIMEK, Arthur; SCHUBERT, Erich. Outlier Detection. In: *Encyclopedia of Database Systems*. Ed. LIU, Ling; ÖZSU, M. Tamer. New York, NY: Springer New York, 2017, s. 1–5. ISBN 978-1-4899-7993-3. Dostupné z DOI: 10.1007/978-1-4899-7993-3_80719-1.
2. HONG, Daocheng; ZHAO, Deshan; ZHANG, Yanchun. The Entropy and PCA Based Anomaly Prediction in Data Streams. *Procedia Computer Science*. 2016-12, roč. 96, s. 139–146. Dostupné z DOI: 10.1016/j.procs.2016.08.115.
3. CANDES, Emmanuel J.; LI, Xiaodong; MA, Yi; WRIGHT, John. *Robust Principal Component Analysis?* 2009. Dostupné z eprint: 0912.3599 (cs.IT).
4. ESKIN, Eleazar; ARNOLD, Andrew; PRERAU, Michael; PORTNOY, Leonid; STOLFO, Salvatore. A Geometric Framework for Unsupervised Anomaly Detection: Detecting Intrusions in Unlabeled Data. *Applications of Data Mining in Computer Security*. 2002-02, roč. 6. ISBN 978-1-4613-5321-8. Dostupné z DOI: 10.1007/978-1-4615-0953-0_4.
5. KUMARI, R.; SHEETANSHU; M., Singh; JHA, R.; N.K., Singh. Anomaly detection in network traffic using K-mean clustering. In: 2016-03, s. 387–393. Dostupné z DOI: 10.1109/RAIT.2016.7507933.
6. RANJAN, Ravi; SAHOO, G. A New Clustering Approach for Anomaly Intrusion Detection. *CoRR*. 2014, roč. abs/1404.2772. Dostupné z arXiv: 1404.2772.
7. GU, Xiaoyi; AKOGLU, Leman; RINALDO, Alessandro. *Statistical Analysis of Nearest Neighbor Methods for Anomaly Detection*. 2019. Dostupné z arXiv: 1907.03813 [stat.ML].
8. SCHÖLKOPF, Bernhard; WILLIAMSON, Robert; SMOLA, Alex; SHAWE-TAYLOR, John; PLATT, John. Support Vector Method for Novelty Detection. In: 1999-01, sv. 12, s. 582–588.
9. LIU, F. T.; TING, K. M.; ZHOU, Z. Isolation Forest. In: *2008 Eighth IEEE International Conference on Data Mining*. 2008, s. 413–422. Dostupné z DOI: 10.1109/ICDM.2008.17.
10. KUNDU, Arnav; SAHU, Abhijeet; SERPEDIN, Erchin; DAVIS, Katherine. A3D: Attention-based auto-encoder anomaly detector for false data injection attacks. *Electric Power Systems Research*. 2020-12, roč. 189, s. 106795. Dostupné z DOI: 10.1016/j.epsr.2020.106795.

11. LUONG, Minh-Thang; PHAM, Hieu; MANNING, Christopher D. *Effective Approaches to Attention-based Neural Machine Translation*. 2015. Dostupné z arXiv: 1508.04025 [cs.CL].

12. QI, Song. Deep Autoencoding Gaussian Mixture Model for Unsupervised Anomaly Detection. 2018-03.

13. YOU, Chen; WANG, Qiwen; SUN, Chao. sBiLSAN:stacked Bidirectional Self-Attention LSTM Network for Anomaly Detection and Diagnosis from System Logs. In: 2019.

14. DEECKE, Lucas; VANDERMEULEN, Robert; RUFF, Lukas; MANDT, Stephan; KLOFT, Marius. Image Anomaly Detection with Generative Adversarial Networks. In: 2018-09.

15. MITA, Takeshi; KANEKO, Toshimitsu; HORI, Osamu. Joint Haar-like features for face detection. In: 2005-11, sv. 2, 1619 –1626 Vol. 2. ISBN 0-7695-2334-X. Dostupné z DOI: 10.1109/ICCV.2005.129.

16. DALAL, Navneet; TRIGGS, Bill. Histograms of Oriented Gradients for Human Detection. *IEEE Conference on Computer Vision and Pattern Recognition (CVPR 2005)*. 2005-06, roč. 2.

17. WANG, Li; HE, Dong-Chen. Texture classification using texture spectrum. *Pattern Recognition*. 1990, roč. 23, č. 8, s. 905–910. ISSN 0031-3203. Dostupné z DOI: https://doi.org/10.1016/0031-3203(90)90135-8.

18. BAY, Herbert; TUYTELAARS, Tinne; VAN GOOL, Luc. SURF: Speeded up robust features. In: 2006-07, sv. 3951, s. 404–417. ISBN 978-3-540-33832-1. Dostupné z DOI: 10.1007/11744023_32.

19. LOWE, David G. Distinctive Image Features from Scale-Invariant Keypoints. *International Journal of Computer Vision*. 2004, roč. 60, č. 2, s. 91–110. ISSN 1573-1405. Dostupné z DOI: 10.1023/B:VISI.0000029664.99615.94.

20. LECUN, Yann; BENGIO, Yoshua; HINTON, Geoffrey. Deep learning. *Nature*. 2015, roč. 521, č. 7553, s. 436–444. ISSN 1476-4687. Dostupné z DOI: 10.1038/nature14539.

21. TOSHEV, Alexander; SZEGEDY, Christian. DeepPose: Human Pose Estimation via Deep Neural Networks. *2014 IEEE Conference on Computer Vision and Pattern Recognition*. 2014. ISBN 9781479951185. Dostupné z DOI: 10.1109/cvpr.2014.214.

22. CAO, Zhe; HIDALGO, Gines; SIMON, Tomas; WEI, Shih-En; SHEIKH, Yaser. *OpenPose: Realtime Multi-Person 2D Pose Estimation using Part Affinity Fields*. 2019. Dostupné z arXiv: 1812.08008 [cs.CV].

23. VARGES DA SILVA, Murilo; NILCEU MARANA, Aparecido. Human action recognition in videos based on spatiotemporal features and bag-of-poses. *Applied Soft Computing*. 2020, roč. 95, s. 106513. ISSN 1568-4946. Dostupné z DOI: https://doi.org/10.1016/j.asoc.2020.106513.

24. *Autoencoder* [online] [cit. 2019-12-25]. Dostupné z: https://iq.opengenus.org/implementing-autoencoder-tensorflow/.

25. BAHDANAU, Dzmitry; CHO, Kyunghyun; BENGIO, Yoshua. *Neural Machine Translation by Jointly Learning to Align and Translate*. 2016. Dostupné z arXiv: 1409.0473 [cs.CL].

26. ELMAN, Jeffrey L. Finding Structure in Time. *Cognitive Science*. 1990, roč. 14, č. 2, s. 179–211. Dostupné z DOI: https://doi.org/10.1207/s15516709cog1402_1.

27. *Understanding LSTM Networks* [online] [cit. 2019-12-25]. Dostupné z: https://colah.github.io/posts/2015-08-Understanding-LSTMs/.

28. *Recurrent Neural Networks* [online] [cit. 2019-12-25]. Dostupné z: http://cs231n.stanford.edu/slides/2017/cs231n_2017_lecture10.pdf.

29. HOCHREITER, Sepp; SCHMIDHUBER, Jürgen. Long Short-term Memory. *Neural computation*. 1997-12, roč. 9, s. 1735–80. Dostupné z DOI: 10.1162/neco.1997.9.8.1735.

30. CHO, Kyunghyun; MERRIENBOER, Bart van; GÜLÇEHRE, Çaglar; BOUGARES, Fethi; SCHWENK, Holger; BENGIO, Yoshua. Learning Phrase Representations using RNN Encoder-Decoder for Statistical Machine Translation. *CoRR*. 2014, roč. abs/1406.1078. Dostupné z arXiv: 1406.1078.

31. HE, Kaiming; ZHANG, Xiangyu; REN, Shaoqing; SUN, Jian. *Deep Residual Learning for Image Recognition*. 2015. Dostupné z arXiv: 1512.03385 [cs.CV].

32. GLOROT, Xavier; BENGIO, Y. Understanding the difficulty of training deep feedforward neural networks. *Journal of Machine Learning Research - Proceedings Track*. 2010-01, roč. 9, s. 249–256.

33. BA, Jimmy Lei; KIROS, Jamie Ryan; HINTON, Geoffrey E. *Layer Normalization*. 2016. Dostupné z arXiv: 1607.06450 [stat.ML].

34. BROWN, Tom B.; MANN, Benjamin; RYDER, Nick; SUBBIAH, Melanie; KAPLAN, Jared; DHARIWAL, Prafulla; NEELAKANTAN, Arvind; SHYAM, Pranav; SASTRY, Girish; ASKELL, Amanda; AGARWAL, Sandhini; HERBERT-VOSS, Ariel; KRUEGER, Gretchen; HENIGHAN, Tom; CHILD, Rewon; RAMESH, Aditya; ZIEGLER, Daniel M.; WU, Jeffrey; WINTER, Clemens; HESSE, Christopher; CHEN, Mark; SIGLER, Eric; LITWIN, Mateusz; GRAY, Scott; CHESS, Benjamin; CLARK, Jack; BERNER, Christopher; MCCANDLISH, Sam; RADFORD, Alec; SUTSKEVER, Ilya; AMODEI, Dario. *Language Models are Few-Shot Learners*. 2020. Dostupné z arXiv: 2005.14165 [cs.CL].